# High-order multigrid strategies for HHO discretizations of elliptic equations

Daniele Di Pietro, Pierre Matalon, Paul Mycek, Ulrich Rüde

▶ **To cite this version:**

Daniele Di Pietro, Pierre Matalon, Paul Mycek, Ulrich Rüde. High-order multigrid strategies for HHO discretizations of elliptic equations. Numerical Linear Algebra with Applications, 2022, 30 (1), pp.e2456. 10.1002/nla.2456 . hal-03531293v2

**HAL Id: hal-03531293**
**https://hal.science/hal-03531293v2**

Submitted on 19 May 2022

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# High-order multigrid strategies for HHO discretizations of elliptic equations*

Daniele A. Di Pietro[†] Pierre Matalon[†‡§¶] Paul Mycek[‡] Ulrich Rüde[‡§]

**Abstract**

This study compares various multigrid strategies for the fast solution of elliptic equations discretized by the Hybrid High-Order method. Combinations of $h$-, $p$- and $hp$-coarsening strategies are considered, combined with diverse intergrid transfer operators. Comparisons are made experimentally on 2D and 3D test cases, with structured and unstructured meshes, and with nested and non-nested hierarchies. Advantages and drawbacks of each strategy are discussed for each case to establish simplified guidelines for the optimization of the time to solution.

**Keywords:** Elliptic partial differential equation, Hybrid High-Order, multigrid, coarsening strategy.

## 1 Introduction

We address in this paper the fast solution of the linear system arising from the Hybrid High-Order (HHO) discretization [9] of scalar elliptic equations. Originally introduced in [10], HHO has been made popular amongst non-conforming methods by its native support of polyhedral elements, its optimal orders of convergence, and its reduced number of degrees of freedom (DoFs) through the static condensation process. In particular, the globally coupled unknowns remaining in the condensed system are located at faces, i.e. supported by the mesh skeleton. This specific structure calls for tailored solvers. Multigrid methods, in particular, require special intergrid transfer operators, inasmuch as the classical ones, used for continuous or discontinuous finite element methods, handle element-defined approximations and are therefore not directly applicable to face-defined ones. Other discretization methods also have globally coupled unknowns located at faces, and adapted multilevel solutions have emerged. Amongst them, one can cite the discontinuous Petrov–Galerkin method [23, 25] and the Hybridizable Discontinuous Galerkin method [17, 21, 26, 31]. Regarding HHO, an $h$-multigrid method was designed by the present authors for the diffusion equation in [12], extended to non-nested mesh hierarchies in [13], and a $p$-multigrid method for the Stokes equation was proposed in [5].

In this work, we focus on high-order approximation. High-order has many advantages in finite element methods. Primarily, for smooth solutions, faster convergence rates with respect to the mesh size allow for the use of a coarser mesh, which limits the memory storage of both the mesh and the matrix, whose global size is usually more dependent on the number of elements than on the number of unknowns per element.The smaller matrix size can also reduce the computational cost of the solution significantly. Additionally, high-order has been shown to be strongly beneficial in the elimination of the *numerical locking* phenomenon [1, 2, 28]. In this context, the design of efficient linear solvers optimized for high-order discretizations proves worthwhile.

In this paper, we propose a comparative study of various multigrid strategies for high-order HHO systems. Three main strategies are explored, characterized by the type of coarsening they implement: (i) $h$-coarsening only, where the mesh is coarsened and the same high polynomial degree is preserved at every level; (ii) $p$-coarsening to lower the degree, followed by $h$-coarsening once the desired low order is reached; (iii) $hp$-coarsening to simultaneously lower the degree and coarsen the mesh, followed by $h$-coarsening once the desired low order is reached. In any case, we assume that the $h$-coarsening phase is able to carry on until

*Funding: ANR project Fast4HHO under contract ANR-17-CE23-0019, jointly with CERFACS
[†]IMAG, Univ. Montpellier, CNRS, Montpellier, France
[‡]CERFACS, Toulouse, France
[§]FAU, Erlangen-Nürnberg, Germany
[¶]Corresponding author (pierre.matalon@gmail.com)

a small enough system is reached. Similar studies have been conducted for the continuous Finite Element method [27], the Discontinuous Galerkin method [4] and for Isogeometric Analysis [30].

This work follows up on [12,13], where an efficient $h$-multigrid method was designed. The convergence of the algorithm, which preserves the polynomial degree at every multigrid level, is shown to be independent of the mesh size (and weakly dependent on the polynomial degree). The intergrid operators defined in those papers shall serve as a basis for the $h$-, $p$- and $hp$-strategies of the present work. In particular, the prolongation operator leverages, in an intermediate step, the higher order potential reconstruction defined by the HHO discretization. This operator allows the gain of one additional degree in the approximation of the error during the intergrid transfer. However, the degree is lowered back to its original value at the end of the prolongation operation in order to preserve the same value at all multigrid levels. Although this trick brings a significant improvement of the convergence rate of the multigrid method (see [12, Section 4.4.1]), lowering back the degree at the end of the prolongation seems to be a waste of possibly valuable higher-order information. One can then legitimately wonder whether the reconstruction of higher degree could be better exploited in a context of $p$-multigrid, where the extra degree could be preserved after prolongation. This idea is at the origin of the alternative strategies discussed in this work. They consist in using the prolongation operator designed for $h$-coarsening also when $p$- and $hp$-coarsening are employed, i.e., when two successive levels do not hold on to the same polynomial degree.

The paper is organized as follows. Section 2 first introduces the diffusion equation with piecewise constant permeability tensor as a model problem, then recalls its HHO discretization and the assembly of the discrete problem. Section 3 defines the multigrid ingredients, especially the interlevel transfer operators used to build the method. The term "interlevel transfer operators" is used to cover both the $h$- and $p$-multigrid cases. In Section 4, we define the high-order strategies we consider. In short, they correspond to the combination of interlevel transfer operators and a global coarsening strategy defining problem reductions with respect to $h$, $p$ and $hp$. In the numerical tests of Section 5, the various strategies are compared on different 2D and 3D test cases with practical, moderate orders of approximation. Structured and unstructured meshes are used, and the multigrid method is applied to hierarchies of nested and non-nested grids. The multigrid method is employed both as a solver and as a preconditioner to a Krylov iteration. Finally, the concluding Section 6 presents guidelines for the best strategy for each test case. In particular, if the setup is neglected, we find that, in the majority of cases, the most efficient one is found amongst the most intuitive strategies, namely, $h$-multigrid only and $p$-multigrid on top of $h$-multigrid at low order. Strategies designed to take advantage of the high-order reconstruction without subsequent degree reduction seem unrewarding in practice. Even though they can sometimes prevail, the gap measured with the other strategies is in general not large enough to be significant. Nonetheless, the strategy involving simultaneous $hp$-coarsening can prove to be useful if the setup cost cannot be neglected. Besides these conclusions, the provided numerical results also offer important insight about the practical computational cost required to solve large scale problems. Indeed, since all strategies yield an optimal solver, the test results can be extrapolated to problems of larger size.

## 2 Model problem and HHO discretization

### 2.1 The diffusion problem

Let $\Omega \subset \mathbb{R}^d$, $d \in \{2,3\}$, be an open, connected, bounded polyhedral domain with boundary $\partial\Omega$. We consider the following boundary value problem: Find $u\colon \Omega \to \mathbb{R}$ such that

$$\begin{cases} -\nabla\cdot(\mathbf{K}\nabla u) = f & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega, \end{cases} \tag{2.1}$$

where $f\colon \Omega \to \mathbb{R}$ denotes the source function and the permeability tensor $\mathbf{K}\colon \Omega \to \mathbb{R}^{d\times d}$ is assumed to be symmetric, uniformly positive definite and piecewise constant over a fixed partition of $\Omega$ into polyhedra.

### 2.2 Mesh definition and notation

Let the couple $\mathcal{M}_h := (\mathcal{T}_h, \mathcal{F}_h)$ define a mesh of the domain $\Omega$, i.e., $\mathcal{T}_h$ is a set of open, polyhedral elements such that $\bigcup_{T\in\mathcal{T}_h} \overline{T} = \overline{\Omega}$, $\mathcal{F}_h$ is the set of element faces, and $h := \max_{T\in\mathcal{T}_h} h_T$ with $h_T$ denoting the diameter of $T$. The mesh is assumed to match the geometrical requirements of [9, Definition 1.4]. $\mathcal{F}_h$ is split into two disjoint subsets: $\mathcal{F}_h^{\mathrm{B}}$, collecting the boundary faces lying on $\partial\Omega$, and $\mathcal{F}_h^{\mathrm{I}}$, collecting the faces located in $\Omega$. For all $T \in \mathcal{T}_h$, we collect into $\mathcal{F}_T$ the faces of $T$. Reciprocally, for all $F \in \mathcal{F}_h$, $\mathcal{T}_F$ collects the elements that

admit $F$ as a face. We also denote by $\mathbf{n}_{TF}$ the unit vector normal to $F$ pointing out of $T$. In the context of problem (2.1), we further assume that the diffusion tensor $\mathbf{K}$ is constant over each element, and we let $\mathbf{K}_T := \mathbf{K}|_T$ for all $T \in \mathcal{T}_h$, as well as $K_{TF} := \mathbf{K}_T\mathbf{n}_{TF} \cdot \mathbf{n}_{TF}$ for all $F \in \mathcal{F}_T$. Finally, for all $X \in \mathcal{T}_h \cup \mathcal{F}_h$, we denote by $(\cdot,\cdot)_X$ the standard inner product of $L^2(X)$ or $L^2(X)^d$.

### 2.3 Local and broken polynomial spaces

The HHO method hinges on discrete unknowns representing polynomial functions local to elements and faces. So, for all $m \in \mathbb{N}_0$ and all $T \in \mathcal{T}_h$ (resp. $F \in \mathcal{F}_h$), we denote by $\mathbb{P}^m(T)$ (resp. $\mathbb{P}^m(F)$) the space spanned by the restriction to $T$ (resp. $F$) of $d$-variate polynomials of total degree $\le m$. From these local polynomial spaces, we can construct the following broken polynomial spaces, respectively supported by the mesh and its skeleton:

$$\mathbb{P}^m(\mathcal{T}_h) := \{v_{\mathcal{T}_h} := (v_T)_{T\in\mathcal{T}_h} \mid v_T \in \mathbb{P}^m(T) \ \ \forall T \in \mathcal{T}_h\},$$
$$\mathbb{P}^m(\mathcal{G}_h) := \{v_{\mathcal{G}_h} := (v_F)_{F\in\mathcal{G}_h} \mid v_F \in \mathbb{P}^m(F) \ \ \forall F \in \mathcal{G}_h\} \text{ for all } \mathcal{G}_h \subset \mathcal{F}_h.$$

Typically, the latter space will be used with $\mathcal{G}_h = \mathcal{F}_h$ and $\mathcal{G}_h = \mathcal{F}_T$ for $T \in \mathcal{T}_h$. For all $X \in \mathcal{T}_h \cup \mathcal{F}_h$, denote by $\pi_X^m \colon L^2(X) \to \mathbb{P}^m(X)$ the $L^2$-orthogonal projector onto $\mathbb{P}^m(X)$ such that for all $v \in L^2(X)$,

$$(\pi_X^m v, w)_X = (v, w)_X \qquad \forall w \in \mathbb{P}^m(X). \tag{2.2}$$

### 2.4 Discrete formulation

Let $k \in \mathbb{N}_0$ be an integer corresponding to the polynomial degree of the scheme. The global and local spaces of *hybrid* variables are respectively defined as

$$\underline{U}_h^k := \left\{\underline{v}_h = (v_{\mathcal{T}_h}, v_{\mathcal{F}_h}) \in \mathbb{P}^k(\mathcal{T}_h) \times \mathbb{P}^k(\mathcal{F}_h)\right\},$$
$$\underline{U}_T^k := \left\{\underline{v}_T = (v_T, v_{\mathcal{F}_T}) \in \mathbb{P}^k(T) \times \mathbb{P}^k(\mathcal{F}_T)\right\} \quad \forall T \in \mathcal{T}_h.$$

For any $\underline{v}_h \in \underline{U}_h^k$, we denote by $\underline{v}_T \in \underline{U}_T^k$ its restriction to $T \in \mathcal{T}_h$. The homogeneous Dirichlet boundary condition is strongly accounted for in the following subspaces:

$$\mathbb{P}^{k,0}(\mathcal{F}_h) := \left\{v_{\mathcal{F}_h} \in \mathbb{P}^k(\mathcal{F}_h) \mid v_F = 0 \ \ \forall F \in \mathcal{F}_h^{\mathrm{B}}\right\}, \qquad \underline{U}_{h,0}^k := \mathbb{P}^k(\mathcal{T}_h) \times \mathbb{P}^{k,0}(\mathcal{F}_h).$$

The global HHO bilinear form associated to the variational formulation of problem (2.1) is $a_h \colon \underline{U}_h^k \times \underline{U}_h^k \to \mathbb{R}$ such that $a_h(\underline{u}_h, \underline{v}_h) := \sum_{T\in\mathcal{T}_h} a_T(\underline{u}_T, \underline{v}_T)$ where, for all $T \in \mathcal{T}_h$, the local bilinear form $a_T \colon \underline{U}_T^k \times \underline{U}_T^k \to \mathbb{R}$ is defined as

$$a_T(\underline{u}_T, \underline{v}_T) := (\mathbf{K}_T\nabla p_T^{k+1}\underline{u}_T, \nabla p_T^{k+1}\underline{v}_T)_T + s_T(\underline{u}_T, \underline{v}_T).$$

In this expression, the first term is responsible for consistency while the second is required to ensure stability of the scheme. The consistency term involves the *local potential reconstruction* $p_T^{k+1} \colon \underline{U}_T^k \to \mathbb{P}^{k+1}(T)$, defined such that, for all $\underline{v}_T \in \underline{U}_T^k$, $p_T^{k+1}\underline{v}_T$ satisfies

$$\begin{cases} (\mathbf{K}_T\nabla p_T^{k+1}\underline{v}_T, \nabla w)_T = -(v_T, \nabla\cdot(\mathbf{K}_T\nabla w))_T + \displaystyle\sum_{F\in\mathcal{F}_T} (v_F, \mathbf{K}_T\nabla w\cdot\mathbf{n}_{TF})_F & \forall w \in \mathbb{P}^{k+1}(T), \quad (2.3a) \\ (p_T^{k+1}\underline{v}_T, 1)_T = (v_T, 1)_T. & (2.3b) \end{cases}$$

Given the local interpolate $\underline{v}_T \in \underline{U}_T^k$ of a function $v \in L^2(\Omega)$, $p_T^{k+1}$ reconstructs an approximation of $v$ of degree $k+1$, i.e., one degree higher than the element or face components. The stabilization bilinear form $s_T$ depends on its argument only through the *difference operators* $\delta_T^k : \underline{U}_T^k \to \mathbb{P}^k(T)$ and $\delta_{TF}^k : \underline{U}_T^k \to \mathbb{P}^k(F)$ for all $F \in \mathcal{F}_T$, respectively defined such that, for all $\underline{v}_T \in \underline{U}_T^k$,

$$\delta_T^k\underline{v}_T := \pi_T^k(p_T^{k+1}\underline{v}_T - v_T) \text{ and } \delta_{TF}^k\underline{v}_T := \pi_F^k(p_T^{k+1}\underline{v}_T - v_F) \text{ for all } F \in \mathcal{F}_T.$$

These operators capture the higher-order correction that the reconstruction $p_T^{k+1}$ adds to the element and face unknowns, respectively. A classical expression for $s_T \colon \underline{U}_T^k \times \underline{U}_T^k \to \mathbb{R}$ is

$$s_T(\underline{v}_T, \underline{w}_T) := \sum_{F\in\mathcal{F}_T} \frac{K_{TF}}{h_F}((\delta_{TF}^k - \delta_T^k)\underline{v}_T, (\delta_{TF}^k - \delta_T^k)\underline{w}_T)_F.$$

The global discrete problem then reads

$$\text{Find } \underline{u}_h \in \underline{U}_{h,0}^k \text{ such that } \ a_h(\underline{u}_h, \underline{v}_h) = \sum_{T\in\mathcal{T}_h} (f, v_T)_T \quad \forall \underline{v}_h \in \underline{U}_{h,0}^k. \tag{2.4}$$

### 2.5 Assembly

The choice of basis functions for the local polynomial spaces defines the local contributions for the assembly of the algebraic problem corresponding to (2.4). For all $T \in \mathcal{T}_h$, those contributions are, respectively for the left- and right-hand side, the matrix $\mathbf{A}_T$ and the vector $\mathbf{B}_T$ such that

$$\mathbf{A}_T := \begin{pmatrix} \mathbf{A}_{TT} & \mathbf{A}_{T\mathcal{F}_T} \\ \mathbf{A}_{\mathcal{F}_T T} & \mathbf{A}_{\mathcal{F}_T\mathcal{F}_T} \end{pmatrix}, \qquad \mathbf{B}_T := \begin{pmatrix} \mathbf{b}_T \\ \mathbf{0} \end{pmatrix}, \tag{2.5}$$

in which the unknowns have been numbered so that element unknowns come first and face unknowns follow. We refer to [9, Appendix B] for further details. Assembly of the local contributions and enforcement of the Dirichlet condition by elimination of the boundary unknowns yield the global linear system

$$\begin{pmatrix} \mathbf{A}_{\mathcal{T}_h\mathcal{T}_h} & \mathbf{A}_{\mathcal{T}_h\mathcal{F}_h^{\mathrm{I}}} \\ \mathbf{A}_{\mathcal{F}_h^{\mathrm{I}}\mathcal{T}_h} & \mathbf{A}_{\mathcal{F}_h^{\mathrm{I}}\mathcal{F}_h^{\mathrm{I}}} \end{pmatrix} \begin{pmatrix} \mathbf{v}_{\mathcal{T}_h} \\ \mathbf{v}_{\mathcal{F}_h^{\mathrm{I}}} \end{pmatrix} = \begin{pmatrix} \mathbf{b}_{\mathcal{T}_h} \\ \mathbf{0} \end{pmatrix}. \tag{2.6}$$

Note that $\mathbf{A}_{\mathcal{T}_h\mathcal{T}_h}$ is block-diagonal, with the blocks $(\mathbf{A}_{TT})_{T\in\mathcal{T}_h}$. It is therefore inexpensive to invert, which is advantageously used in the so-called process of *static condensation*. Indeed, the element unknowns can be expressed in terms of the face unknowns in the first equation of (2.6), yielding

$$\mathbf{v}_{\mathcal{T}_h} = -\mathbf{A}_{\mathcal{T}_h\mathcal{T}_h}^{-1}\mathbf{A}_{\mathcal{T}_h\mathcal{F}_h^{\mathrm{I}}}\mathbf{v}_{\mathcal{F}_h^{\mathrm{I}}} + \mathbf{A}_{\mathcal{T}_h\mathcal{T}_h}^{-1}\mathbf{b}_{\mathcal{T}_h}, \tag{2.7}$$

i.e.,

$$\mathbf{v}_T = -\mathbf{A}_{TT}^{-1}\mathbf{A}_{T\mathcal{F}_T}\mathbf{v}_{F_T} + \mathbf{A}_{TT}^{-1}\mathbf{b}_T \qquad \forall T \in \mathcal{T}_h, \tag{2.8}$$

where $\mathbf{v}_T$ denotes the restriction to $T$ of $\mathbf{v}_{\mathcal{T}_h}$, and $\mathbf{v}_{F_T}$ the restriction of $\mathbf{v}_{\mathcal{F}_h^{\mathrm{I}}}$ to the faces of $T$ interior to the domain, completed by zeros for boundary faces. Finally, replacing $\mathbf{v}_{\mathcal{T}_h}$ with its expression (2.7) in the second equation of (2.6) gives the so-called *trace* or *condensed* system

$$\widetilde{\mathbf{A}}\mathbf{v}_{\mathcal{F}_h^{\mathrm{I}}} = \widetilde{\mathbf{b}} \quad \text{where} \quad \widetilde{\mathbf{A}} := \mathbf{A}_{\mathcal{F}_h^{\mathrm{I}}\mathcal{F}_h^{\mathrm{I}}} - \mathbf{A}_{\mathcal{F}_h^{\mathrm{I}}\mathcal{T}_h}\mathbf{A}_{\mathcal{T}_h\mathcal{T}_h}^{-1}\mathbf{A}_{\mathcal{T}_h\mathcal{F}_h^{\mathrm{I}}}, \quad \widetilde{\mathbf{b}} := -\mathbf{A}_{\mathcal{F}_h^{\mathrm{I}}\mathcal{T}_h}\mathbf{A}_{\mathcal{T}_h\mathcal{T}_h}^{-1}\mathbf{b}_{\mathcal{T}_h}. \tag{2.9}$$

Solving (2.9) for the face unknowns is the costliest operation and the purpose of our multigrid methods, after which the element unknowns can be locally and independently recovered via (2.8).

## 3 Multigrid ingredients

### 3.1 Multilevel hierarchy

Assume a multigrid hierarchy of $L$ levels indexed by $\ell \in \{1, \dots, L\}$ and sorted so that $L$ refers to the finest level and 1 to the coarsest one. For each level $\ell \in \{1, \dots, L\}$, denote by $h_\ell$ the corresponding mesh size. For the sake of simplicity, the quantities so far subscripted by $h$ are now simply subscripted by $\ell$ instead of $h_\ell$, so we denote by $\mathcal{M}_\ell := (\mathcal{T}_\ell, \mathcal{F}_\ell)$ instead of $\mathcal{M}_{h_\ell} := (\mathcal{T}_{h_\ell}, \mathcal{F}_{h_\ell})$ the mesh at level $\ell$. Finally, denote by $k_\ell$ the HHO polynomial degree at level $\ell$. Relative to those levels, consider a hierarchy of (not necessarily nested) polyhedral meshes $(\mathcal{M}_\ell)_{\ell\in\{1,\dots,L\}}$ satisfying the requirements of Section 2.2. Considering two successive levels $\ell$ (fine) and $\ell-1$ (coarse), the coarsening principles impose that $k_{\ell-1} \le k_\ell$ and $h_{\ell-1} \ge h_\ell$. Specifically, we distinguish three coarsening strategies:

- $k_{\ell-1} < k_\ell$ and $\mathcal{M}_{\ell-1} = \mathcal{M}_\ell$ ($p$-coarsening),
- $k_{\ell-1} = k_\ell$ and $h_{\ell-1} > h_\ell$ ($h$-coarsening),
- $k_{\ell-1} < k_\ell$ and $h_{\ell-1} > h_\ell$ ($hp$-coarsening).

We further assume that mesh coarsening not only involves the coarsening of the elements, but also that of the faces; see [12, Section 4.4.3] for details and justifications.

### 3.2 Interlevel transfer operators

In this section, we consider two successive levels $\ell$ (fine) and $\ell-1$ (coarse), and define various prolongation and restriction operators.

#### 3.2.1 Prolongation by decondensation (for $h$-, $p$- and $hp$-coarsening)

The prolongation operator employed in our previous works [12, 13] in the context of $h$-multigrid is here extended to the cases where $k_\ell > k_{\ell-1}$ and/or $\mathcal{M}_{\ell-1} = \mathcal{M}_\ell$, in order to handle $p$- and $hp$-coarsening as well. Its philosophy and construction remain unchanged, and hinge on a sequence of operations: decondensation of the cell unknowns to reconstruct a potential in the cells from values on the coarse faces; increase of the polynomial degree via the higher-order reconstruction operator; transfer to the fine mesh; trace on the fine faces. This sequence is formalized by the definition of the following operators:

- $\Theta_\ell^k\colon \mathbb{P}^k(\mathcal{F}_\ell) \to \mathbb{P}^{k+1}(\mathcal{T}_\ell)$ for all $\ell$ and $k$:
  Let $v_{\mathcal{F}_\ell} \in \mathbb{P}^k(\mathcal{F}_\ell)$ denote the argument of $\Theta_\ell^k$. For all $T \in \mathcal{T}_\ell$, let us denote by $v_{\mathcal{F}_T} := (v_F)_{F\in\mathcal{F}_T}$ the restriction of $v_{\mathcal{F}_\ell}$ to $T$, and by $\mathbf{v}_{\mathcal{F}_T} := (\mathbf{v}_F)_{F\in\mathcal{F}_T}$ its algebraic representation as vectors of coefficients in the chosen polynomial bases. With these notations, we define the algebraic volumic potential $\mathbf{v}_T$ by reversing the static condensation, i.e. via (2.8) without the constant term:
  $$\mathbf{v}_T := -\mathbf{A}_{TT}^{-1}\mathbf{A}_{T\mathcal{F}_T}\mathbf{v}_{\mathcal{F}_T}. \tag{3.1}$$
  Once the potential $v_T \in \mathbb{P}^k(T)$ is retrieved from (3.1) via its algebraic representation, we improve its accuracy through the higher-order reconstruction operator $p_T^{k+1}$ defined by (2.3). The local contribution to $\Theta_\ell^k$ is therefore given by
  $$(\Theta_\ell^k v_{\mathcal{F}_\ell})_{|T} := p_T^{k+1}(v_T, v_{\mathcal{F}_T}). \tag{3.2}$$
- $J_{\ell-1,\ell}^k\colon \mathbb{P}^k(\mathcal{T}_{\ell-1}) \to \mathbb{P}^k(\mathcal{T}_\ell)$ for all $\ell$ and $k$:
  This operator handles the transfer from the coarse to the fine cells without altering the polynomial degree. If the meshes are nested, then $J_{\ell-1,\ell}^k$ is merely defined as the natural injection operator, induced by the embedding of the function spaces. In particular, if both meshes are identical ($p$-coarsening), it reduces to the identity. If the meshes are non-nested, we define $J_{\ell-1,\ell}^k$ as the $L^2$-orthogonal projector onto $\mathbb{P}^k(\mathcal{T}_\ell)$.
- $\Pi_\ell^{k,k'}\colon \mathbb{P}^k(\mathcal{T}_\ell) \to \mathbb{P}^{k',0}(\mathcal{F}_\ell)$ for all $\ell, k, k'$:
  Let $v := (v_T)_{T\in\mathcal{T}_\ell} \in \mathbb{P}^k(\mathcal{T}_\ell)$ denote the argument of $\Pi_\ell^{k,k'}$. Let $F \in \mathcal{F}_\ell$ and, if $F \in \mathcal{F}_\ell^{\mathrm{I}}$, let $T_1, T_2$ denote the distinct elements in $\mathcal{T}_F \subset \mathcal{T}_\ell$. The local contribution of $F$ corresponds to a weighted average of the traces of degree $k'$ computed on both sides of $F$:
  $$(\Pi_\ell^{k,k'} v)_{|F} := \begin{cases} w_{T_1F}\,\pi_F^{k'}(v_{T_1|F}) + w_{T_2F}\,\pi_F^{k'}(v_{T_2|F}) & \text{if } F \in \mathcal{F}_\ell^{\mathrm{I}}, \\ 0 & \text{otherwise,} \end{cases} \tag{3.3}$$
  with the weighting values $w_{T_iF} := \frac{K_{T_iF}}{K_{T_1F}+K_{T_2F}}$ for $i \in \{1,2\}$. Note that, if $k \le k'$, $\pi_F^{k'}$ corresponds to the natural injection. The degree is actually reduced only if $k > k'$.

We refer the reader to [12, 13] for more details about these operators. The *prolongation operator by decondensation* $P\colon \mathbb{P}^{k_{\ell-1},0}(\mathcal{F}_{\ell-1}) \to \mathbb{P}^{k_\ell,0}(\mathcal{F}_\ell)$ is defined as
$$P := \Pi_\ell^{k_{\ell-1}+1,k_\ell} \circ J_{\ell-1,\ell}^{k_{\ell-1}+1} \circ \Theta_{\ell-1}^{k_{\ell-1}}. \tag{3.4}$$
The definition of $P$ is represented in Figure 1. In this composition, we start at the coarse level $\ell-1$, from face-defined polynomials of degree $k_{\ell-1}$. The application of $\Theta_{\ell-1}^{k_{\ell-1}}$ results in a broken cell-defined polynomial of degree $k_{\ell-1}+1$. $J_{\ell-1,\ell}^{k_{\ell-1}+1}$ transfers that broken polynomial to the fine mesh $\mathcal{M}_\ell$ (possibly equal to $\mathcal{M}_{\ell-1}$, in which case $J_{\ell-1,\ell}^{k_{\ell-1}+1}$ is the identity). Finally, $\Pi_\ell^{k_{\ell-1}+1,k_\ell}$ defines polynomials of degree $k_\ell$ on the fine faces. Notice that, in the case of $p$- or $hp$-coarsening, $k_{\ell-1}+1 \le k_\ell$, so no degree reduction is performed in (3.3). The use of this operator with $p$- and $hp$-coarsening then gives rise to novel strategies in which, unlike with $h$-coarsening, no information is lost in the degree reduction. These strategies are discussed in Section 4.

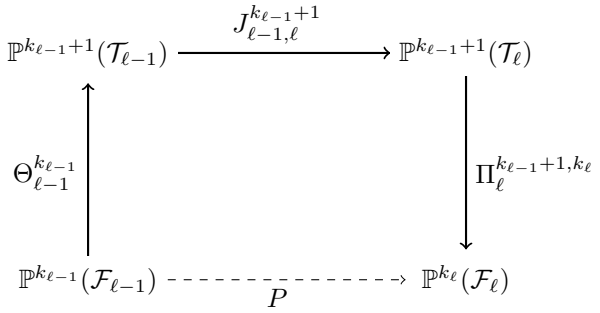



Figure 1: The prolongation operator $P$ by decondensation.

#### 3.2.2 Prolongation by natural injection (for $p$-coarsening only)

In $p$-coarsening, we work with the same mesh at both levels so that, in particular, $\mathcal{F}_{\ell-1} = \mathcal{F}_\ell$. Then, $k_{\ell-1} < k_\ell$ implies $\mathbb{P}^{k_{\ell-1}}(F) \subset \mathbb{P}^{k_\ell}(F)$ for all $F \in \mathcal{F}_{\ell-1}$, which justifies the natural injection operators

$$I_F^{k_{\ell-1},k_\ell} : \mathbb{P}^{k_{\ell-1}}(F) \ni v \mapsto v \in \mathbb{P}^{k_\ell}(F) \qquad \forall F \in \mathcal{F}_{\ell-1}. \tag{3.5}$$

The global prolongation operator by natural injection $I^{k_{\ell-1},k_\ell} : \mathbb{P}^{k_{\ell-1}}(\mathcal{F}_{\ell-1}) \to \mathbb{P}^{k_\ell}(\mathcal{F}_\ell)$ is locally defined for all $v \in \mathbb{P}^{k_{\ell-1}}(\mathcal{F}_{\ell-1})$ by

$$(I^{k_{\ell-1},k_\ell} v)_{|F} := I_F^{k_{\ell-1},k_\ell}(v_{|F}).$$

#### 3.2.3 Restriction by adjoint (for $h$-, $p$- and $hp$-coarsening)

The restriction operator is defined as the adjoint of the prolongation operator with respect the coarse and fine skeleton-defined inner products. Consequently, the matrix representation of the restriction is the transpose of the matrix representation of the prolongation.

#### 3.2.4 Restriction by $L^2$-orthogonal projection (for $p$-coarsening only)

$p$-coarsening implies $\mathcal{F}_{\ell-1} = \mathcal{F}_\ell$, and we denote by $\pi^{k_\ell,k_{\ell-1}} : \mathbb{P}^{k_\ell}(\mathcal{F}_\ell) \to \mathbb{P}^{k_{\ell-1}}(\mathcal{F}_{\ell-1})$ the $L^2$-orthogonal projector defined such that for all $F \in \mathcal{F}_{\ell-1}$ and all $v \in \mathbb{P}^{k_\ell}(\mathcal{F}_\ell)$, $(\pi^{k_\ell,k_{\ell-1}} v)_{|F} := \pi_F^{k_{\ell-1}}(v_{|F})$, with $\pi_F^{k_{\ell-1}}$ verifying (2.2). It is interesting to notice that $\pi^{k_\ell,k_{\ell-1}}$ is the adjoint operator of the natural injection $I^{k_{\ell-1},k_\ell}$ w.r.t. the $L^2$-inner product on the mesh skeleton. Indeed, for all $F \in \mathcal{F}_\ell$,

$$(\pi_F^{k_{\ell-1}} v, w)_F \overset{(2.2)}{=} (v, w)_F \overset{(3.5)}{=} (v, I_F^{k_{\ell-1},k_\ell} w)_F \qquad \forall (v, w) \in \mathbb{P}^{k_\ell}(F) \times \mathbb{P}^{k_{\ell-1}}(F).$$

### 3.3 Smoothers

In order to relax together the set of unknowns related to the same local polynomial, block versions of standard fixed-point methods are used. Block Jacobi, block SOR and their derivatives are typical choices. In our numerical tests, we use block Gauss–Seidel. The block size corresponds to the number of degrees of freedom per face, which is given, for each level $\ell$, by

$$\dim(\mathbb{P}^{k_\ell}(F)) = \frac{(k_\ell + d - 1)!}{k_\ell!\,(d-1)!}.$$

Notice that the block size depends on the multigrid level and the coarsening method. If level $\ell-1$ is obtained from level $\ell$ by $p$-coarsening, then the block size at level $\ell$ is relative to $k_\ell$ and the block size at level $\ell-1$ is relative to $k_{\ell-1} < k_\ell$. Considering a 3D example with $k_\ell = 3$ and $k_{\ell-1} = 1$, the block sizes shall be 10 at level $\ell$ and 3 at level $\ell-1$. On the contrary, if only $h$-coarsening is used, then $k_\ell$ remains constant across the multigrid levels, and so does the block size.

### 3.4 Efficient implementation with hierarchical, $L^2$-orthogonal bases

As pointed out in [5], the implementation of the $p$-multigrid part of the algorithm can be optimized by using modal, hierarchical, and $L^2$-orthogonal local polynomial bases on the faces. Assume $k_\ell > k_{\ell-1}$ and

$\mathcal{M}_{\ell-1} = \mathcal{M}_\ell$. Let $F \in \mathcal{F}_\ell$, and let $\mathcal{B}_F^{k_\ell}$ be a hierarchical, orthogonal basis of $\mathbb{P}^{k_\ell}(F)$. The hierarchical structure allows us to extract from $\mathcal{B}_F^{k_\ell}$ the basis functions of degree at most $k_{\ell-1}$ to obtain a basis $\mathcal{B}_F^{k_{\ell-1}}$ of $\mathbb{P}^{k_{\ell-1}}(F)$ with the same properties. Let $n_{k_\ell} := \mathrm{card}(\mathcal{B}_F^{k_\ell})$ and $n_{k_{\ell-1}} := \mathrm{card}(\mathcal{B}_F^{k_{\ell-1}})$. The following efficient implementations are then possible:

1. *Natural injection:* given $v \in \mathbb{P}^{k_{\ell-1}}(F)$ and $\mathbf{v} \in \mathbb{R}^{n_{k_{\ell-1}}}$ the vector of coefficients in the decomposition of $v$ on the modal basis $\mathcal{B}_F^{k_{\ell-1}}$, the algebraic representation of $I_F^{k_{\ell-1},k_\ell} v$ is obtained by simply extending $\mathbf{v}$ to $\mathbb{R}^{n_{k_\ell}}$, via the padding of $(n_{k_\ell} - n_{k_{\ell-1}})$ zero coefficients. This operation can then be applied dynamically without any storage or preliminary construction in a setup phase. Note that this implementation only hinges on the basis being hierarchical, orthogonality is not requested.

2. *$L^2$-orthogonal projection:* reciprocally, given $v \in \mathbb{P}^{k_\ell}(F)$ and $\mathbf{v} \in \mathbb{R}^{n_{k_\ell}}$ the vector of coefficients in the decomposition of $v$ on the modal basis $\mathcal{B}_F^{k_\ell}$, the algebraic representation of $\pi_F^{k_{\ell-1}} v$ is obtained by shrinking $\mathbf{v}$ to a size of $n_{k_{\ell-1}}$, i.e. by discarding the coefficients corresponding to the degrees $> k_{\ell-1}$. Like the preceding operator, the projection can then also be applied on the fly, without setup. This implementation hinges on the orthogonality of the basis functions, and is easy to prove via the orthogonal direct sum $\mathbb{P}^{k_\ell}(F) = \mathbb{P}^{k_{\ell-1}}(F) \bigoplus \mathrm{span}(\mathcal{B}_F^{k_\ell} \setminus \mathcal{B}_F^{k_{\ell-1}})$.

3. *Assembly of the coarse matrix:* instead of resorting to the rediscretization of the equation at the order $k_{\ell-1}$, for the same reasons as for the $L^2$-orthogonal projection, it suffices to discard the high-order coefficients in the matrix of level $\ell$. It corresponds to extracting the top-left corner of size $n_{k_{\ell-1}} \times n_{k_{\ell-1}}$ from each original block of size $n_{k_\ell} \times n_{k_\ell}$. One can remark that the matrices of both the natural injection and the $L^2$-orthogonal projection are rectangular identity matrices (one being the transpose of the other), so, in the case where they are both used as transfer operators, the coarse matrix also corresponds to the Galerkin operator. The two-grid scheme is then in the configuration of the variational framework, ensuring convergence.

### 3.4.1 Practical implementation of suitable polynomial bases

First of all, recall that the above optimizations require only the *face* polynomial bases to be hierarchical and orthogonal. In principle, the bases for the elements do not have to be. However, in the presence of distorted elements, non-orthogonal bases may dramatically affect the condition number of the matrix. In this case, or if high precision is required from the approximation, using orthogonal bases for the elements is also recommended.

An orthogonal basis can be locally constructed by applying the Modified Gram-Schmidt (MGS) algorithm (w.r.t. the local $L^2$-inner product) to an initial hierarchical basis composed of Cartesian products of local monomials. Note that MGS also performs normalization, which means that the basis functions are scaled according to the size of the element in which they are defined. This can, in fact, raise numerical issues in the simultaneous presence of large and comparatively small elements, which typically happens when the mesh is locally refined. Although we find that the normalization has little effect on the condition number, the scaling of the basis functions is strongly affecting the convergence properties of our multigrid method in the presence of aggressive local refinement. To maintain a scaling of the basis functions that is suited for the multigrid algorithm, we employ MGS without the normalization step.

The local orthogonalization of the bases induces additional computational cost. Besides the actual orthogonalization process, the computation of the integrals during assembly requires more work. Indeed, the basis functions are defined as linear combinations of the non-orthogonal, initial polynomials. Consequently, in an implementation with quadrature, the evaluation of a basis function on a quadrature point implies the evaluation of all the initial polynomials occurring in the linear combination. That said, the cost can be reduced significantly with more efficient quadrature-free methods. We especially refer to [11, Section 5.3]. Nonetheless, to avoid extra cost, one should use an already orthogonal basis whenever possible. Especially, the Legendre basis is $L^2$-orthogonal on the unit interval. It can then be used on the faces of 2D problems (edges) and, in the form of tensor products, on shapes that can be mapped to Cartesian reference shapes. It typically includes, in 2D, quadrilateral elements, and in 3D, quadrilateral faces and hexahedral elements.

# 4 High-order multigrid strategies

## 4.1 Multigrid algorithms under consideration

Suppose a fixed hierarchy of $M$ meshes, which will serve for the implementation of $(M-1)$ $h$-coarsening operations. Note that the term of $h$-coarsening does not refer to how the mesh hierarchy is actually built. It refers in an abstract way to the relation linking a fine mesh to its immediately coarser one in the hierarchy. Consequently, we speak of $h$-coarsening between two meshes even if the fine one was obtained by refinement of the coarse one. We assume that the problem assembled on the coarsest mesh is small enough for the linear system to be solved directly, regardless of the polynomial degree. We then consider three types of strategies according to the successive coarsening operations they perform to build the coarse multigrid levels, starting from the fine problem:

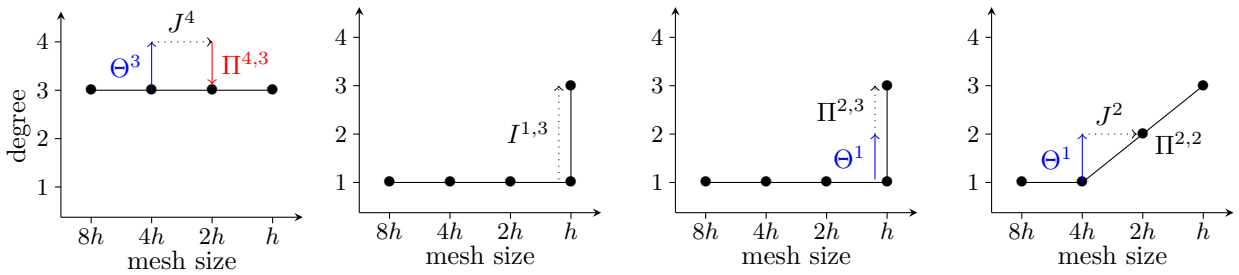
- "$h$ only": one multigrid level is built for each mesh in the hierarchy, meaning that $M$ levels are built, linked by $(M-1)$ $h$-coarsening operations. The original polynomial degree is conserved at every level.
- "$p$ then $h$": first, $p$-coarsening is successively performed to construct a first set of levels. Once the polynomial degree has been reduced to $k=1$, the mesh hierarchy is used to build further coarse levels.
- "$hp$ then $h$": first, $hp$-coarsening is used to simultaneously lower the degree and coarsen the mesh. Then, assuming that $k=1$ is reached before exhausting the mesh hierarchy, the remaining meshes are used to construct further coarse levels. This assumption is usually verified in practice for large problems and moderate values of the original polynomial degree.

In what follows, $h$-coarsening is realized by successively doubling the mesh size. Concerning $p$-coarsening, multiple $p$-multigrid algorithms have shown that it is not necessary to lower the degree by only one at each step. Decreasing the degree more aggressively [4,5,14,18,27], or even going directly to the low order in one single $p$-coarsening step [24], often proves to be more efficient. In the present multigrid method, we have empirically identified that the best performance with $p$-multigrid seems to be achieved when lowering the degree by two. In case of "$p$ then $h$" coarsening, we then enforce $k_{\ell-1} := \max\{k_\ell - 2, 1\}$ for the $p$- part. However, in case of the naturally more aggressive $hp$-coarsening, we only decrement the degree by one at each step, i.e. $k_{\ell-1} := k_\ell - 1$.

When two successive levels do not share the same mesh, i.e., in case of $h$- or $hp$-coarsening, we use the prolongation $P$ constructed by decondensation (cf. Section 3.2.1). The associated restriction operator, in this case, is its adjoint (cf. Section 3.2.3). Otherwise, when the levels are linked by $p$-coarsening, we explore two possibilities: (i) the prolongation by natural injection $I^{k_{\ell-1},k_\ell}$ (cf. Section 3.2.2) along with the restriction by $L^2$-orthogonal projection $\pi^{k_\ell,k_{\ell-1}}$ (cf. Section 3.2.4); (ii) the prolongation by decondensation $P$ along with its adjoint. These settings define four global strategies, summarized in Table 1. Note that not all possible combinations are considered. We have retained only those that can ensure the symmetry of the multigrid operator. The strategies of Table 1 are represented in Figure 2 on an example with $k=3$ and a hierarchy of four meshes.

| Strategy name | Coarsening | Degree reduction | $p$-coarsening prolong. | $p$-coarsening restrict. | $h$- or $hp$-coarsening prolong. | $h$- or $hp$-coarsening restrict. |
|---|---|---|---|---|---|---|
| `h-only` | $h$ only | 0 | | | $P$ | $P^\top$ |
| `p-h` | $p$ then $h$ | 2 | $I^{k_{\ell-1},k_\ell}$ | $\pi^{k_\ell,k_{\ell-1}}$ | $P$ | $P^\top$ |
| `p-h*` | $p$ then $h$ | 2 | $P$ | $P^\top$ | $P$ | $P^\top$ |
| `hp-h` | $hp$ then $h$ | 1 | | | $P$ | $P^\top$ |

Table 1: High-order multigrid strategies. The column "Coarsening" refers to the coarsening algorithm followed to successively construct the coarse multigrid levels from the fine one. The third column quantifies the "Degree reduction" between two successive levels $\ell$ and $\ell-1$ linked by $p$- or $hp$-coarsening: the value corresponds to $m$ in the enforced relation $k_{\ell-1} := \max\{k_\ell - m, 1\}$. The remaining columns on the right indicate the prolongation and restriction operators used in case of $p$-coarsening on one hand, and $h$- or $hp$-coarsening on the other hand.




(a) `h-only` (b) `p-h` (c) `p-h*` (d) `hp-h`

Figure 2: The multigrid strategies of Table 1, given a fine problem with $k = 3$ and a hierarchy of four meshes. The black, connected dots correspond to the multigrid levels, the uppermost rightmost one representing the finest level. The most relevant prolongation operators are also plotted. As such, the operator $P$ is decomposed according to the suboperators in (3.4), where the subscripts related to levels have been omitted for readability. Superscripts, related to degrees, are however conserved. Blue arrows raise the degree, red ones decrease it, and dotted black ones have no effect on the degree.

### 4.2 Interpretation

While `h-only` and `p-h` (see Table 1) correspond to the most classical and intuitive options, `p-h*` and `hp-h` are more original. Both these alternatives are designed to circumvent the loss of information induced by the use of the higher-order reconstruction in the operator $P$ in the original `h-only` strategy. Indeed, if the reconstruction of polynomials in the cells (performed by $\Theta_{\ell-1}^{k_{\ell-1}}$) raises the degree to $k_{\ell-1}+1$, it is lowered back to $k_{\ell-1}$ in the trace operation on the fine faces (performed by $\Pi_{\ell}^{k_{\ell-1}+1,k_\ell}$, where $k_\ell = k_{\ell-1}$); see Figure 2a. Although this trick ultimately improves the accuracy of the trace of degree $k_{\ell-1}$, one might wonder whether preserving the higher degree $k_{\ell-1}+1$ on the trace could be more efficient. This supposes that the fine level be associated to a degree greater or equal to $k_{\ell-1}+1$. If so, either the fine level also has a finer mesh, yielding the strategy `hp-h` (see Figure 2d), or it has the same mesh (i.e. $p$-coarsening only is enforced), yielding `p-h*` (see Figure 2c).

We can also interpret `p-h*` with respect to `p-h`. In the $p$ part of `p-h`, each local polynomial of degree $k_{\ell-1}$ is simply injected identically into a larger polynomial space, namely, of degree $k_{\ell-1}+2$ (recall that in our configuration, a reduction of 2 degrees is performed between level $\ell$ and level $\ell-1$); see Figure 2b. This injection leaves the burden of solving for the two degrees separating the levels to the fine grid smoother. On the other hand, the operator $P$ that is used in `p-h*` instead of the natural injection actually offers the computation of one degree of approximation higher. Thus, here, only one polynomial degree needs to be handled by the fine smoother. However, $P$ does not deliver on the fine level an identical representation of the coarse function, which may have negative consequences. The numerical experiments of Section 5 indeed show that `p-h*` is usually less efficient than `p-h`.

`hp-h` can be similarly interpreted with respect to `h-only`. While in `h-only`, the reconstructed higher degree is lowered back, it can be conserved in `hp-h`, which may counterbalance the elevated aggressiveness of the *hp*-coarsening. The numerical tests of Section 5 show that `hp-h` sometimes prevails over `h-only`.

## 5 Numerical tests

### 5.1 Experimental setup and implementation

The multigrid strategies described in Section 4 and summarized in Table 1 are used to solve the condensed linear system (2.9), either as a solver or as a preconditioner for a Krylov method. The smoother used at every level is the block Gauss–Seidel method, with block size corresponding to the number of unknowns per face (cf. Section 3.3). Following the recommendations of [12, Section 4.2.2], we use V-cycle with post-smoothing only; see also [19, 20]. This choice has been found to optimize the performance of the multigrid methods (in terms of computational work and execution time) for all considered strategies. In particular, we use V(0,3) in 2D and V(0,6) in 3D. These unsymmetrical cycles prevent the use of the standard conjugate gradient (CG) as outer iteration. Instead, such a nonsymmetric preconditioner may be used in a *flexible*

conjugate gradient (FCG), as justified in [6]. We stress that we obtain significantly better performance with our unsymmetrical cycles and FCG than with symmetrical cycles and CG. We then choose FCG (see [22] for the detailed algorithm) as outer Krylov method for the use of our multigrid method as a preconditioner. Denoting by $\mathbf{r}_i$ the algebraic residual of (2.9) at iteration $i$ and $||\cdot||_2$ the Euclidean norm on the space of coefficients, the solver stops when $||\mathbf{r}_i||_2/||\widetilde{\mathbf{b}}||_2$ is lower than a given tolerance.

Our software uses shared-memory parallelism with a maximum of 24 parallel threads. Parallel execution is employed in the assembly and the setup phase of the algorithm, since here only local computations are performed. Specifically, this applies to the following operations: (i) the assembly of the coarse matrices obtained by rediscretization of the problem on the coarse meshes in the context of $h$- and $hp$-coarsening; (ii) the extraction of the coarse matrices from the fine one when performing $p$-coarsening using orthogonal bases; (iii) the assembly of the prolongation operators $P$ (3.4); (iv) the factorization of the diagonal blocks of the operators for future use in the block Gauss–Seidel iterations. Note that, assuming orthogonal bases, the prolongation by natural injection and the restriction by $L^2$-orthogonal projection do not need any setup (cf. Section 3.4).

The block Gauss–Seidel smoother, however, is an inherently sequential algorithm. We have therefore chosen to adhere to the sequential ordering to avoid the situation that the convergence rates could depend on the specific parallelization strategy. This is necessary, since the main purpose of this current study is to compare high-order solution strategies independently of parallelization issues. To this end, we have defined metrics of computational cost that do not depend on the specific implementation and parallelism: theoretical computational work and number of iterations. As a complement to this, we will also cite CPU times.

The theoretical computational work is expressed in *work units* (WUs), where one WU amounts to the number of flops involved in one application of the fine grid matrix (namely, twice the number of non-zeros in the matrix). The number of WUs can be interpreted, for instance, as the cost corresponding to the same number of Richardson iterations. This notion of a WU is commonly used in the multigrid literature to express the performance of an algorithm in light of the *textbook multigrid efficiency* [7,29]. In particular, the textbook paradigm quantifies the unspecified constant in the optimality property. According to Brandt [7], an ideal multigrid algorithm should solve a system with an algebraic error below the discretization error for a computational cost no higher than 10 WUs.

### 5.2 Square domain, Cartesian mesh

To establish a baseline, let us start with the square domain $\Omega := [0,1]^2$, discretized by a uniform, Cartesian mesh. The tensor $\mathbf{K}$ is the identity matrix. The source function $f$ is computed with respect to the exact solution $u\colon [0,1]^2 \ni (x,y) \mapsto \sin(4\pi x)\sin(4\pi y)$. The problem is discretized by the HHO method with $k = 5$. At this high degree, a mesh of $128 \times 128$ square elements is enough to achieve an $L^2$-error lower than $10^{-12}$, close to machine precision. The mesh hierarchy is composed of four embedded Cartesian meshes. Orthogonal Legendre polynomials are chosen as local polynomial bases for both cells and faces, which allows the optimizations described in Section 3.4.

The multigrid method is employed as a solver to solve the condensed linear system (2.9), using each of the strategies of Table 1. The tolerance defining the stopping criterion is set to $10^{-12}$, in accordance with the discretization error. Figure 7 presents bar plots that compare the performance of the various strategies. In (a), the computational work of the iteration phase is expressed in equivalent number of WUs. In (b), this work is measured in terms of CPU time. In (c), the convergence is evaluated in terms of the number of iterations to reach convergence. In (d), the setup cost is measured in CPU time. Table (e) gives additional information about the multigrid method, namely, the number of levels built and the asymptotic convergence rate $\varrho$, defined as the geometric mean of the residual convergence ratios of the last five iterations:

$$\varrho := \sqrt[5]{\frac{||\mathbf{r}_n||_2}{||\mathbf{r}_{n-5}||_2}},$$

$n$ denoting the last iteration. Finally, plot (f) stresses the robustness of the convergence of the methods with respect to the mesh size.

First, one can see that the results of the iteration phase in computational work and CPU time are consistent, yielding the same ranking. In particular, `p-h` (in red) is found to be the most computationally efficient strategy, with about half the theoretical cost of the `h-only` strategy (in blue). Clearly, this is due to

its better convergence rate (see plot (c)). Second, plot (d) shows that `p-h` also has the cheapest setup, due to the fact that (i) the levels issued from the $p$-coarsening operations do not need rediscretization (the coarse matrices are extracted from the fine one), (ii) the three rediscretizations performed after the $h$-coarsenings are made at the low order $k = 1$, and (iii), the transfer operators between levels related by $p$-coarsening do not require any setup. In comparison, the `h-only` strategy rediscretizes the problem three times on the coarse meshes at the high order $k = 5$, which is costlier, and the respective prolongation operators $P$ also have to be built at that high order. Next, one can remark that the strategy `p-h*` does not seem to pay off. One could have expected the operator $P$, here used for $p$-multigrid, to have a greater impact. Indeed, while the natural injection leaves the burden of solving the two polynomial degrees separating the levels to the fine smoother, the operator $P$ actually offers the computation of one degree of approximation higher, leaving only one remaining degree to be handled by the fine smoother.

Although the lowest possible discretization error is already achieved, the mesh is refined multiple times to increase the problem size in order to exhibit the asymptotic behaviour of the solver. The flat curves of Figure 7f clearly show the optimality of each method, the convergence rates of which are independent of the number of unknowns.

Figure 8 presents the analogous information as Figure 7, this time using the multigrid method as a preconditioner for FCG. Note that the number of iterations displayed here corresponds to the iterations of the preconditioned FCG solver. One can see that the strategy rankings of Figure 7 are preserved and that the overall computational cost is reduced. In particular, the linear system is solved to machine precision with a computational work corresponding to about 65 WUs. This is significantly higher than what is expected for classical *textbook multigrid efficiency* (TME), which suggests that 10 WUs should suffice to solve the linear system. However, e.g. [29, p. 318] states that, in many cases, solvers require an order of magnitude higher cost than the ideal goal of 10 WUs. In our case, additional difficulties arise because of the high order of the discretization, the face-based nature of the system, and the correspondingly more complex matrix structure. This makes it unclear whether TME with 10 WUs could be achieved. Note also that 10 WUs are only claimed for multigrid when used in a full multigrid setting (i.e. a nested iteration), when the initial guess is recursively computed by multigrid employed on a coarser level. In our current experiments, we only use V-cycles that are insufficient to reach TME. We point out, however, that TME has been attained, e.g., in [16] for conforming 3D discretizations.

When the exact solution is sufficiently smooth, the use of high polynomial degrees reduces the cost to achieve a desired precision. Figure 3a presents, for each value of $k \in \{2, 3, 4, 5\}$, how finely the unit square must be decomposed by Cartesian elements (and how large the resulting linear system must be) to achieve an $L^2$-error $< 10^{-12}$. Clearly, the higher the polynomial degree, the coarser the mesh can be, and the smaller the linear system becomes. Figure 3b illustrates the time spent solving each of those problems, including initial assembly and setup, using the FCG solver preconditioned by the multigrid method with strategy `p-h`. It shows that, for approximations resulting in equivalent $L^2$-errors, high orders have a clear advantage over low ones in terms of global time to solution. As a consequence, if the solution is smooth, raising the order of approximation is more efficient to improve the solution quality than refining the mesh, but of course this requires that a suitable and efficient solver is available. If the solution is non-smooth, the higher orders cannot be fully exploited and their effect is mitigated (cf. Section 5.4 and Figure 5). A classical solution consists in resorting to posteriori-driven adaptivity.

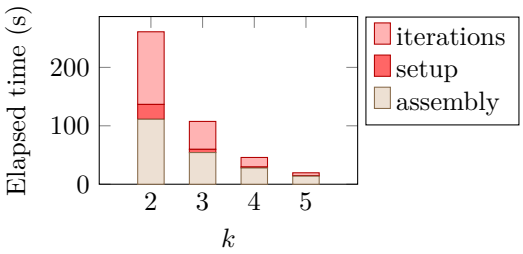
### 5.3 Square domain, unstructured triangular mesh

Using the problem settings of the preceding test case, the square domain is now discretized by a coarse Delaunay triangulation. The initial triangulation is refined until the mesh is fine enough for the approximate solution to achieve an $L^2$-error lower than $10^{-12}$. Each refinement step consists in subdividing each coarse triangle into four fine triangles connecting the edge middle points. Setting $k = 5$, the fine mesh is composed of about 11,000 elements, for a system size of about 100,000 rows. The mesh hierarchy obtained (three meshes) is used to model the operations of $h$-coarsening in the multigrid method. The polynomial bases are chosen as the orthogonal Legendre basis on the faces (here, edges) and, in the elements, local monomials are orthogonalized element-wise (cf. Section 3.4.1).

Figure 9 presents the numerical results of the different strategies when the multigrid method is used as a solver. The ranking, here, differs from that of the preceding test case with Cartesian meshes, The change of element type seems to favour the strategies prioritizing mesh coarsening, namely, `h-only` and `hp-h`. Note

| $k$ | Mesh size | Elements | Unknowns |
|---|---|---|---|
| 2 | $h$ | $1024^2$ | 6,285,312 |
| 3 | $2h$ | $512^2$ | 2,093,056 |
| 4 | $4h$ | $256^2$ | 652,800 |
| 5 | $8h$ | $128^2$ | 195,072 |

(a) For each value of $k$, Number of Cartesian elements and respective number of unknowns in the system




(b) Time to achieve equivalent $L^2$-errors for various values of $k$, w.r.t. the discretizations of (a)

Figure 3: Table (a) presents, for each value of $k$, how fine the unit square must be refined by Cartesian elements (and how large the linear system must be) to achieve an $L^2$-error $< 10^{-12}$. Plot (b) shows, for each of those problems, the time spent on initial assembly, setup of the multigrid solver (strategy `p-h`), and solution of the system using the preconditioned FCG.

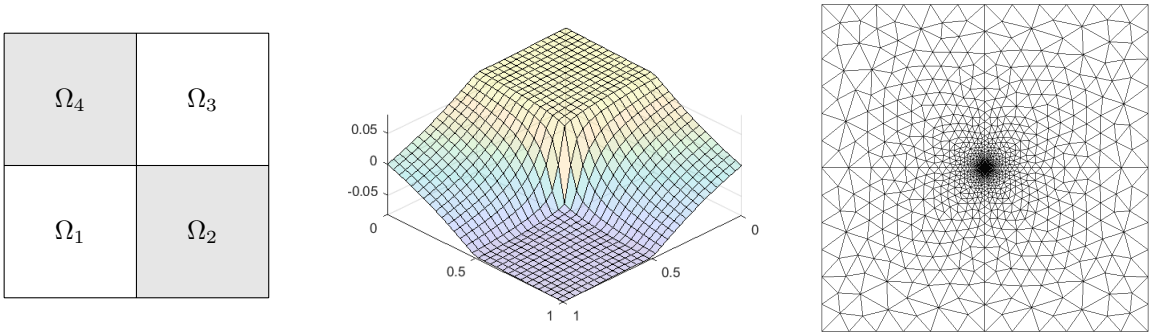
that they are also the ones with the costliest setup, due to the rediscretizations at high orders on the coarse meshes. This makes them competitive only when the system needs to be solved many times. If the system needs to be solved with one or few right-hand sides, then strategies leveraging $p$-coarsening have a clear advantage. Moreover, one can remark the higher cost of the setup in comparison with Cartesian elements, owing to the dynamic orthogonalization of the local polynomial bases.

Figure 10 presents the results of the multigrid method as a preconditioner. Note that the ranking given by the computational work and the CPU time ((a) and (b), respectively) do not concur. For this reason, we avoid drawing conclusions and recall the bias these results are subject to. First, the intergrid transfers of the $p$-multigrid part of `p-h` perform no theoretical flops, although their execution does consume CPU time, which makes the computational work of `p-h` more optimistic than its CPU time. This remark is nonetheless to be mitigated by the fact that, in `p-h`, the intergrid transfers consumes less than 2% of the CPU time. The second bias is the dependency of the CPU time on the implementation, contrary to the theoretical computational work, which is objective. On the other hand, the CPU time usually reflects the practical time to solution more accurately. Following this criterion, the ranking of the strategies follows that of the multigrid used as a solver, i.e. a preference for `h-only` and `hp-h`.

### 5.4 Kellogg heterogeneous problem

The Kellogg problem is a well-known heterogeneous diffusion benchmark problem. The square domain is partitioned into four quadrants $(\Omega_i)_{i=1,\dots,4}$ such as in Figure 4a, and the diffusion coefficient $\mathbf{K}$ is such that $\mathbf{K}_{|\Omega_1} = \mathbf{K}_{|\Omega_3} = \kappa_1\mathbf{I}$ and $\mathbf{K}_{|\Omega_2} = \mathbf{K}_{|\Omega_4} = \kappa_2\mathbf{I}$, where $\mathbf{I}$ is the identity matrix and $\kappa_1, \kappa_2$ are positive scalar values. The source function is $f = 0$. The discontinuities in the coefficient reduce the regularity of the solution, which is $H^{1+\epsilon}$, $0 < \epsilon \leq 1$. This low regularity makes it a difficult problem, often used as a benchmark for adaptive element methods. Indeed, the exact solution is known. In our case, we follow [15, Example 5.4] and refer to it for the solution's analytical formula. In particular, the heterogeneity ratio is $\kappa_1/\kappa_2 \approx 161$, inducing $\epsilon = 0.1$ and leading to the solution plotted in Figure 4b. Non-homogeneous Dirichlet conditions are enforced on the boundary of the domain.

While the preceding test cases could be solved at machine precision with very few unknowns, the low regularity of the Kellogg solution requires a finer mesh, corresponding to a larger problem size. In particular, we use a hierarchy built by successive refinements of an unstructured mesh locally refined around the central singularity (see Figure 4c), and set $k = 5$. The fine mesh has over 350,000 elements, producing over 3 million unknowns. The $L^2$-error reached with this setting is about $3.10^{-3}$. Figure 11 presents the numerical results of the various high-order multigrid strategies to solve the system (cf. Table 1). Given the discretization error, a tolerance of $10^{-3}$ would be sufficient. However, to stress the convergence of the algorithm, we set it to $10^{-12}$. The rankings match those of the preceding test case (unit square with unstructured triangular mesh), presenting `h-only` and, to a lesser extent, `hp-h`, as the most suitable alternatives if the setup can be neglected. This tends to strengthen the hypothesis that unstructured triangular meshes favour strategies that prioritize $h$-coarsening. In particular, `h-only` has clearly the best asymptotic convergence rate (0.27




(a) Heterogeneity pattern (b) Analytical solution (c) Locally refined mesh

Figure 4: Kellogg problem.

versus about 0.40 or more for the other strategies; see (e)).

Figure 12 presents the results of the multigrid methods as preconditioners to FCG. As in the preceding test case, the rankings provided by the computational work (a) and the iteration CPU time (b) are not consistent, so we avoid drawing conclusions. That said, following the CPU time criterion, our implementation favors again `h-only` and `hp-h` for use as preconditioners. Recall also that, surprisingly, while `p-h*` was more efficient than `p-h` as a solver, it is the other way around when these strategies are used as preconditioners.

Figure 3 has shown the clear advantage of high order on the overall time to solution when the exact solution is smooth. For the sake of comparison, Figure 5 reproduces the experiment on the Kellogg problem, i.e. when the solution is non-smooth. Figure 5a presents, for each polynomial degree $k \in \{1, 2, 3, 4, 5\}$, the coarsest mesh required (still obtained from successive refinements of an initial coarse mesh such as Figure 4c), as well as the corresponding number of unknowns, to achieve an $L^2$-error smaller than $5 \times 10^{-3}$. As there is no distorted or stretched element, we simply use the monomial bases in cells and on faces to save up the cost of the orthogonalization in this experiment. Figure 3b presents the overall time to solution (including initial assembly and multigrid setup) to solve those problems. The system is solved using the `h-only` strategy as a preconditioner (found to be the most efficient solver in computational time, setup excluded, for this problem; see discussion above). The tolerance is set to $10^{-3}$, in accordance with the discretization error. As a consequence, the solver converges in less than four iterations, which makes this step almost negligible. The results here are more difficult to interpret than on the smooth test case because the $L^2$-error is hard to reduce, which means that a small decrease of the error may actually demand a lot of additional work. Consequently, as the problems of Figure 5a do not generate exactly the same $L^2$-errors, comparing their computational costs under the assumption that they achieve equivalent $L^2$-errors is somewhat erroneous. However, this setting is enough to roughly illustrate the limitations of high orders. Considering the approximate solutions yielded by all problems of equivalent quality, the problem with $k = 3$ is a sweet spot, and a better option than $k = 4$ on the same mesh or $k = 5$ on a mesh with twice the mesh size. Furthermore, noticing that problems with $k = 1$ and $k = 4$ in fact generate the same discretization error, one can see that it is more efficient to choose $k = 1$ on a fine mesh than $k = 4$ on a coarser one.

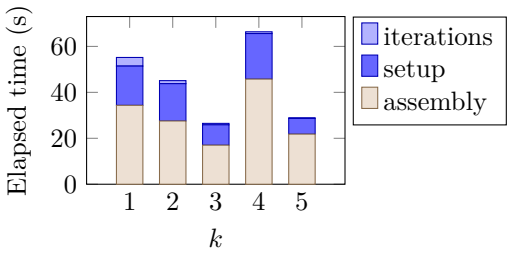
### 5.5 Cubic domain, Cartesian mesh

$\Omega := [0, 1]^3$, discretized by an uniform, Cartesian mesh. The tensor $\mathbf{K}$ is homogeneous and isotropic. The source function $f$ is computed with respect to the manufactured solution $u\colon [0, 1]^3 \ni (x, y, z) \mapsto \sin(4\pi x)\sin(4\pi y)\sin(4\pi z)$. The problem is discretized by the HHO method with $k = 3$ on a mesh composed of $64^3$ cubic elements, generating about 8 million unknowns. With this configuration, the approximation achieves an $L^2$-error of $3.10^{-7}$. The mesh hierarchy is composed of five embedded Cartesian meshes. Orthogonal Legendre polynomials are chosen as local polynomial bases for the cells and faces, which allows the optimizations described in Section 3.4.

Figure 13 (resp. Figure 14) presents the results of the multigrid method used as a solver (resp. as a preconditioner), with the various high-order strategies of Table 1. Clearly, as in 2D, the $p$-multigrid approach prevails with Cartesian meshes.

| $k$ | Mesh size | Unknowns | $L^2$-error |
|---|---|---|---|
| 1 | $h$ | 1,065,728 | $4.45 \times 10^{-3}$ |
| 2 | $2h$ | 399,552 | $4.68 \times 10^{-3}$ |
| 3 | $4h$ | 133,120 | $4.77 \times 10^{-3}$ |
| 4 | $4h$ | 166,400 | $4.45 \times 10^{-3}$ |
| 5 | $8h$ | 49,872 | $4.75 \times 10^{-3}$ |

(a) For each value of $k$, Number of Cartesian elements and respective number of unknowns in the system




(b) Time to achieve equivalent $L^2$-errors for various values of $k$, w.r.t. the discretizations of (a)

Figure 5: Table (a) presents for the Kellogg problem and for each value of $k$, how fine the unit square must be discretized by Cartesian elements (and how large the linear system must be) to achieve an $L^2$-error lower than $5 \times 10^{-3}$. Plot (b) shows, for each of those problems, the time spent on initial assembly, setup of the multigrid solver (strategy `p-h`), and solution of the system using the preconditioned FCG.

### 5.6 Cubic domain, structured tetrahedral mesh

This test case follows the settings of the preceding one, except for the mesh type, which is now tetrahedral. The elements are arranged in a structured manner: the domain is uniformly subdivided in a Cartesian way, where each cube is itself subdivided into six tetrahedra through the Kuhn triangulation (see [3, Figures 8 and 9]). This decomposition allows an easy and practical mesh coarsening, in the sense that halving the number of cubes in the Cartesian decomposition results in a coarse mesh that embeds the fine one and doubles its mesh size. Additionally, both meshes comprise the same tetrahedral shapes, therefore ensuring the same aspect ratios. We build this way a hierarchy of five nested meshes, each one composed of $6N^3$ tetrahedra, $N \in \{32, 16, 8, 4, 2\}$. With $k = 3$, the arising linear system contains about 4 million unknowns and the approximation generates an error of $5.10^{-6}$. The monomials are locally orthogonalized to allow the optimizations described in Section 3.4.

When the multigrid method is used as a solver, the results of Figure 15 favor the `h-only` strategy, agreeing with the similar test case in 2D (cf. Section 5.3). As a preconditioner (Figure 16), `h-only` is still favored by the CPU times of the iterations, although to a lesser extent.

### 5.7 3D complex domain, non-nested meshes

This test case is the complex geometry of a Geneva wheel with unstructured tetrahedral mesh, represented in Figure 6. The source $f$ is a piecewise polynomial function. The main difficulty of this test case lies in the construction of the mesh hierarchy. Indeed, proceeding by tetrahedral refinement of an initial coarse mesh degrades the aspect ratio of the elements, causing bad performances of our multigrid method, highly sensitive to the mesh quality (cf. [12, Section 4.5]). The use of non-nested meshes circumvents the problem. We then use independent Delaunay triangulations of the domain, choosing the mesh sizes so that each pair of successive meshes enforces a geometric coarsening ratio close to two. In order to avoid prohibitive computational costs, the $L^2$-orthogonal projection $J^k_{\ell-1,\ell}$, introduced for all $k \geq 0$ in Section 3.2.1, is computed approximately via the method described in [13, Section 3.3]. In particular, the strict definition of $J^k_{\ell-1,\ell}$ involves the evaluation of the geometric intersections between all overlapping fine and coarse elements, which induces complex and costly computations. Instead, the intersection of a coarse element $T_c \in \mathcal{T}_{\ell-1}$ and an overlapping fine one $T_f \in \mathcal{T}_\ell$ is approximated in the following manner:

$$T_c \cap T_f \approx \bigcup \{t \in \mathrm{Sub}(T_f) \text{ s.t. } \mathrm{barycenter}(t) \in T_c\}, \tag{5.1}$$

where $\mathrm{Sub}(T_f)$ is a fixed subdivision of $T_f$, obtained in practice by tetrahedral refinement. With this method, the $L^2$-inner product between a coarse basis function and a fine one attached to the overlapping element $T_f$ is decomposed into a sum of integrals over tetrahedra, namely, those whose barycenter is included in $T_c$. The numerical tests in [13] showed that one Bey's refinement is enough to construct an approximate $L^2$-orthogonal projection operator accurate enough to be successfully used in our multigrid method up to $k = 2$. From $k = 3$, two refinement steps are necessary. However, given that Bey's method decomposes a

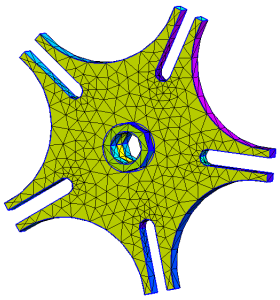


Figure 6: Tetrahedral mesh of a Geneva wheel.

tetrahedron into eight smaller ones, the computational work required to assemble the operator is multiplied by eight for each additional refinement. As we find two refinements too costly in practice when integrals are computed by quadrature rules, we limit ourselves to $k = 2$ for the strategies `h-only` and `hp-h`. For higher orders, strategies lowering the degree before dealing with the non-nested meshes are recommended.

Information regarding the hierarchy of non-nested meshes is provided in Figure 17f. The rest of Figure 17 presents the results of the various strategies of multigrid method used as a solver. Interestingly, `hp-h` is found to be the most efficient. As a preconditioner, following Figure 18, `hp-h` is still favored by the CPU times of the iteration step. One can remark the high computational cost compared to the other problems. This can be explained by (i) the very use of the $L^2$-orthogonal projection, which is an approximation in itself (even computed exactly) and therefore implies a loss of precision with respect to the initial function it is applied to; (ii) compared to nested meshes, the stencil of the prolongation is enlarged, thus increasing the cost of the grid transfers: indeed, while in the case of nested meshes, each fine element is included in only one coarse element, a non-nested fine element usually overlaps multiple coarse elements, thus multiplying the size of the corresponding stencil.

## 6 Conclusion

In this paper, we defined and compared several high-order multigrid strategies (cf. Table 1) for the fast solution of elliptic equations discretized by the HHO method. Assuming that the integrals are computed with quadrature rules, the setup phase can become time-consuming in high-order and, according to the strategy, even take a significant part of the computations, especially if the bases have to be locally orthogonalized. Consequently, if the arising linear system needs to be solved with only one right-hand side, then the usual method consisting in plugging a $p$-multigrid on top of an $h$-one (strategy `p-h`) clearly stands out, owing to its very cheap setup. Otherwise, if the system is to be solved with many right-hand sides, the best option may differ according to the space dimension and the type of mesh. The most important take-away of this study is that the best choice may significantly reduce the time to solution as compared to other strategies. This remark should encourage users to put different strategies to the test in order to find out the most efficient one for their specific case study. Based on our numerical experiments, we can, nonetheless, attempt to draw simplified conclusions to help in this process. In particular, the strategy `p-h*`, especially designed to exploit the higher-order reconstruction without loss of information, yields disappointing results while being also the least intuitive strategy. It can therefore be discarded in practice. `hp-h`, also designed to exploit the higher-order reconstruction, often shows performances close to `h-only` while having a much cheaper setup. In 2D, while Cartesian meshes see `p-h` prevailing, `h-only` and `hp-h` are favored in more general cases. In 3D, on the other hand, the results are not so clear. Nonetheless, it is to be noted that in complex cases, where non-nested meshes and approximate $L^2$-orthogonal projections are employed, setup computations are severely more demanding at high-order, rendering `h-only` hardly practicable. This conclusion could be modified by the use of more efficient strategies for the numerical computation of integrals such as the technique of [8].

The best strategy for the method used as a solver is also often the one prevailing when the method is used as a preconditioner. Even if the multigrid method is already optimal as a solver, the preconditioning technique allows to save, on average, about 25% of the computational work.

Another meaningful remark is that the computational work required exhibits large variations according to the type of mesh. Typically, a solution on an unstructured mesh may demand twice the work of that on

a Cartesian mesh, mainly due to an increased number of iterations.

The methods of this article will have to be significantly extended when applied to other settings, such as, e.g., strongly anisotropic equations or Stokes flow. We refer to the literature for the general adaptations needed in multigrid solvers for these problem classes. Studying such extensions specifically for the $h$-/$p$-multigrid methods of this article, will be the topic of future research.

## Conflict of interest

This study does not have any conflict to disclose.

## References


[1] I. Babuška and M. Suri. Locking effects in the finite element approximation of elasticity problems. *Numerische Mathematik*, 62(1):439–463, 1992. doi:10.1007/BF01396238.

[2] I. Babuška and M. Suri. On Locking and Robustness in the Finite Element Method. *SIAM Journal on Numerical Analysis*, 29(5):1261–1293, 1992. doi:10.1137/0729075.

[3] J. Bey. Tetrahedral grid refinement. *Computing*, 55(4):355–378, 1995. doi:10.1007/BF02238487.

[4] L. Botti, A. Colombo, A. Crivellini, and M. Franciolini. $h$-$p$-$hp$-Multilevel discontinuous Galerkin solution strategies for elliptic operators. *International Journal of Computational Fluid Dynamics*, 33(9):362–370, 2019. doi:10.1080/10618562.2019.1688306.

[5] L. Botti and D. A. Di Pietro. $p$-Multilevel preconditioners for HHO discretizations of the Stokes equations with static condensation. *Communications on Applied Mathematics and Computation*, 2021. doi:10.1007/s42967-021-00142-5.

[6] H. Bouwmeester, A. Dougherty, and A. V. Knyazev. Nonsymmetric Preconditioning for Conjugate Gradient and Steepest Descent Methods 1. *Procedia Computer Science*, 51:276–285, 2015. doi:10.1016/j.procs.2015.05.241.

[7] A. Brandt. Barriers to Achieving Textbook Multigrid Efficiency (TME) in CFD. *Institute for Computer Applications in Science and Engineering, NASA Langley Research Center*, 1998.

[8] E. B. Chin, J. B. Lasserre, and N. Sukumar. Numerical integration of homogeneous functions on convex and nonconvex polygons and polyhedra. *Comput. Mech.*, 56(6):967–981, 2015. doi:10.1007/s00466-015-1213-7.

[9] D. A. Di Pietro and J. Droniou. *The Hybrid High-Order method for polytopal meshes*. Number 19 in Modeling, Simulation and Application. Springer International Publishing, 2020. doi:10.1007/978-3-030-37203-3.

[10] D. A. Di Pietro and A. Ern. A hybrid high-order locking-free method for linear elasticity on general meshes. *Comput. Meth. Appl. Mech. Engrg.*, 283:1–21, 2015. doi:10.1016/j.cma.2014.09.009.

[11] D. A. Di Pietro, L. Formaggia, and R. Masson, editors. *Polyhedral Methods in Geosciences*, volume 27 of *SEMA SIMAI Springer Series*. Springer International Publishing, Cham, 2021. doi:10.1007/978-3-030-69363-3.

[12] D. A. Di Pietro, F. Hülsemann, P. Matalon, P. Mycek, U. Rüde, and D. Ruiz. An $h$-multigrid method for Hybrid High-Order discretizations. *SIAM Journal on Scientific Computing*, 43(5):S839–S861, 2021. doi:10.1137/20M1342471.

[13] D. A. Di Pietro, F. Hülsemann, P. Matalon, P. Mycek, U. Rüde, and D. Ruiz. Towards robust, fast solutions of elliptic equations on complex domains through hybrid high-order discretizations and non-nested multigrid methods. *International Journal for Numerical Methods in Engineering*, 122(22):6576–6595, 2021. doi:10.1002/nme.6803.

[14] N. Fehn, P. Munch, W. A. Wall, and M. Kronbichler. Hybrid multigrid methods for high-order discontinuous Galerkin discretizations. *Journal of Computational Physics*, 415:109538, 2020. doi:10.1016/j.jcp.2020.109538.

[15] H. Guo and X. Yang. Gradient recovery for elliptic interface problem: I. body-fitted mesh. *Communications in Computational Physics*, 23(5):1488–1511, 2018. doi:10.4208/cicp.OA-2017-0026.

[16] N. Kohl and U. Rüde. Textbook efficiency: massively parallel matrix-free multigrid for the Stokes system. *arXiv:2010.13513 [cs, math]*, 2020. To appear in SIAM J. Sci. Comput.

[17] P. Lu, A. Rupp, and G. Kanschat. Homogeneous multigrid for HDG. *IMA Journal of Numerical Analysis*, (drab055), 2021. URL: 10.1093/imanum/drab055, doi:10.1093/imanum/drab055.

[18] B. S. Mascarenhas, B. T. Helenbrook, and H. L. Atkins. Coupling p-multigrid to geometric multigrid for discontinuous galerkin formulations of the convection–diffusion equation. *Journal of Computational Physics*, 229(10):3664–3674, 2010. doi:10.1016/j.jcp.2010.01.020.

[19] A. Miraçi, J. Papež, and M. Vohralík. A Multilevel Algebraic Error Estimator and the Corresponding Iterative Solver with p-Robust Behavior. *SIAM Journal on Numerical Analysis*, 58(5):2856–2884, 2020. doi:10.1137/19M1275929.

[20] A. Miraçi, J. Papež, and M. Vohralík. A-Posteriori-Steered p-Robust Multigrid with Optimal Step-Sizes and Adaptive Number of Smoothing Steps. *SIAM Journal on Scientific Computing*, 43(5):S117–S145, 2021. doi:10.1137/20M1349503.

[21] S. Muralikrishnan, T. Bui-Thanh, and J. N. Shadid. A multilevel approach for trace system in HDG discretizations. *Journal of Computational Physics*, 407:109240, 2020. doi:10.1016/j.jcp.2020.109240.

[22] Y. Notay. Flexible Conjugate Gradients. *SIAM Journal on Scientific Computing*, 22(4):1444–1460, 2000. doi:10.1137/S1064827599362314.

[23] S. Petrides and L. Demkowicz. An adaptive multigrid solver for DPG methods with applications in linear acoustics and electromagnetics. *Computers & Mathematics with Applications*, 87:12–26, 2021. doi:10.1016/j.camwa.2021.01.017.

[24] P. Rasetarinera and M. Y. Hussaini. An efficient implicit discontinuous spectral galerkin method. *Journal of Computational Physics*, 172(2):718–738, 2001. doi:10.1006/jcph.2001.6853.

[25] N. V. Roberts and J. Chan. A geometric multigrid preconditioning strategy for dpg system matrices. *Computers & Mathematics with Applications*, 74(8):2018–2043, 2017.

[26] J. Schütz and V. Aizinger. A hierarchical scale separation approach for the hybridized discontinuous Galerkin method. *Journal of Computational and Applied Mathematics*, 317:500–509, 2017. doi:10.1016/j.cam.2016.12.018.

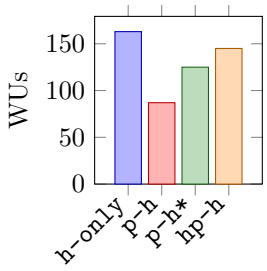
[27] H. Sundar, G. Stadler, and G. Biros. Comparison of multigrid algorithms for high-order continuous finite element discretizations. *Numerical Linear Algebra with Applications*, 22(4):664–680, 2015. doi:10.1002/nla.1979.

[28] M. Suri. Analytical and computational assessment of locking in the hp finite element method. *Computer Methods in Applied Mechanics and Engineering*, 133(3):347–371, 1996. doi:10.1016/0045-7825(95)00947-7.

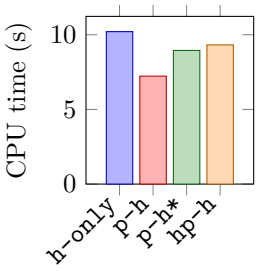
[29] J. L. Thomas, B. Diskin, and A. Brandt. Textbook Multigrid Efficiency for Fluid Simulations. *Annual Review of Fluid Mechanics*, 35(1):317–340, 2003. doi:10.1146/annurev.fluid.35.101101.161209.

[30] R. Tielen, M. Möller, D. Göddeke, and C. Vuik. p-multigrid methods and their comparison to h-multigrid methods within Isogeometric Analysis. *Computer Methods in Applied Mechanics and Engineering*, 372:113347, 2020. doi:10.1016/j.cma.2020.113347.

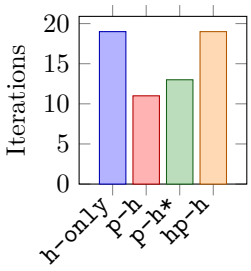
[31] T. Wildey, S. Muralikrishnan, and T. Bui-Thanh. Unified Geometric Multigrid Algorithm for Hybridized High-Order Finite Element Methods. *SIAM Journal on Scientific Computing*, 41(5):S172–S195, 2019. doi:10.1137/18M1193505.



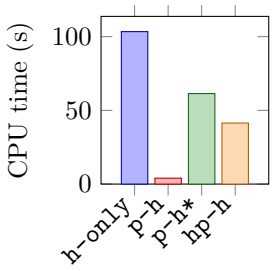

(a) Iteration comput. work




(b) Iteration time



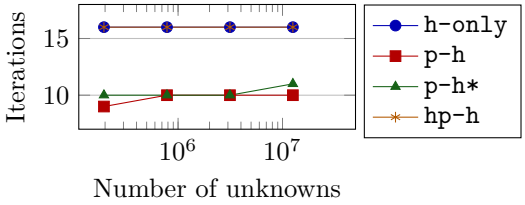

(c) Number of iterations




(d) Setup time

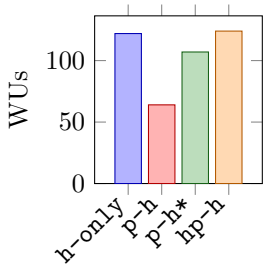
| Strategy | Coarsening | Levels | it | $\varrho$ |
|---|---|---|---|---|
| `h-only` | 3 $h$- | 4 | 19 | 0.27 |
| `p-h` | 2 $p$-, 3 $h$- | 6 | 11 | 0.09 |
| `p-h*` | 2 $p$-, 3 $h$- | 6 | 13 | 0.15 |
| `hp-h` | 3 $hp$- | 4 | 19 | 0.27 |

(e) Additional information: type and number of coarsening operations performed, number of levels built, number of iterations (it), asymptotic convergence rate ($\varrho$).

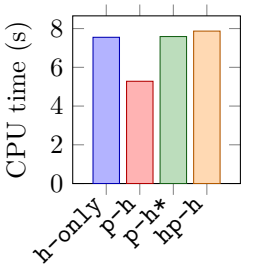



(f) Asymptotic behaviour. (Due to the accumulation of rounding errors in large problems, the residual reduction reaches a floor, so the tolerance is set to $10^{-10}$.)

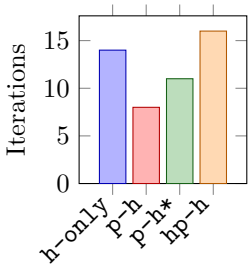
Figure 7: **Unit square, Cartesian mesh** $128 \times 128$, $k = 5$. The multigrid method is used **as a solver** with the strategies of Table 1. Convergence is assumed achieved once the backward error reaches the tolerance of $10^{-12}$.



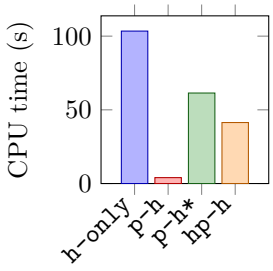

(a) Iteration comput. work




(b) Iteration time




(c) Number of iterations

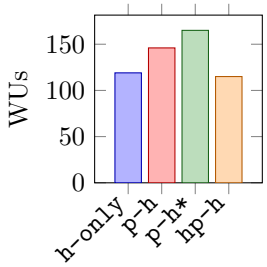



(d) Setup time

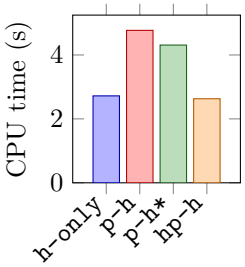
| Strategy | Coarsening | Levels | it | $\varrho$ |
|---|---|---|---|---|
| `h-only` | 3 $h$- | 4 | 14 | 0.16 |
| `p-h` | 2 $p$-, 3 $h$- | 6 | 8 | 0.03 |
| `p-h*` | 2 $p$-, 3 $h$- | 6 | 11 | 0.11 |
| `hp-h` | 3 $hp$- | 4 | 16 | 0.20 |

(e) Additional information: type and number of coarsening operations performed, number of levels built, number of iterations of preconditioned FCG (it), asymptotic convergence rate ($\varrho$).

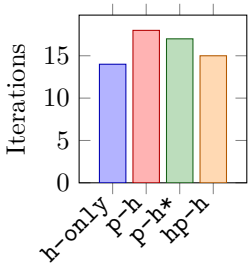
Figure 8: **Unit square, Cartesian mesh** $128 \times 128$, $k = 5$. The multigrid method is used **as a preconditioner to FCG** with the strategies of Table 1. Convergence is assumed achieved once the backward error reaches the tolerance of $10^{-12}$.



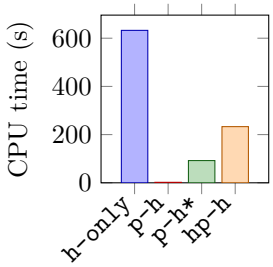

(a) Iteration comput. work




(b) Iteration time



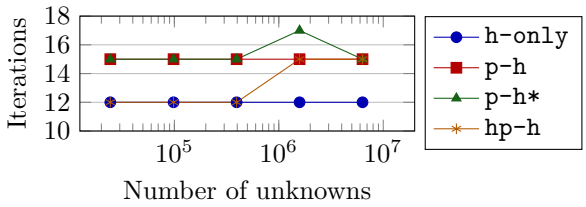

(c) Number of iterations




(d) Setup time

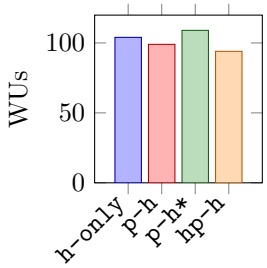
| Strategy | Coarsening | Levels | it | $\varrho$ |
|---|---|---|---|---|
| `h-only` | 2 $h$- | 3 | 14 | 0.17 |
| `p-h` | 2 $p$-, 2 $h$- | 5 | 18 | 0.22 |
| `p-h*` | 2 $p$-, 2 $h$- | 5 | 17 | 0.21 |
| `hp-h` | 2 $hp$- | 3 | 15 | 0.19 |

(e) Additional information: type and number of coarsening operations performed, number of levels built, number of iterations (it), asymptotic convergence rate ($\varrho$).

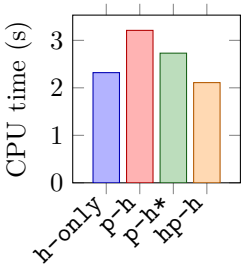



(f) Asymptotic behaviour. (Due to the accumulation of rounding errors in large problems, the residual reduction reaches a floor, so the tolerance is set to $10^{-10}$.)

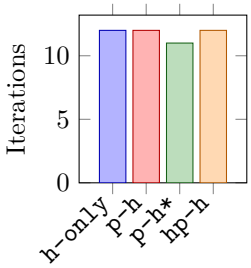
Figure 9: **Unit square, unstructured triangular mesh**, $k = 5$. The multigrid method is used as a **solver** with the strategies of Table 1. Convergence is assumed achieved once the backward error reaches the tolerance of $10^{-12}$.



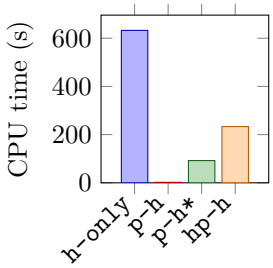

(a) Iteration comput. work




(b) Iteration time




(c) Number of iterations

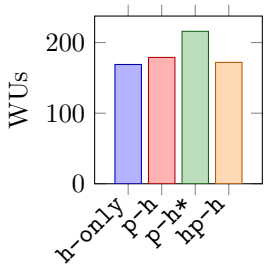



(d) Setup time

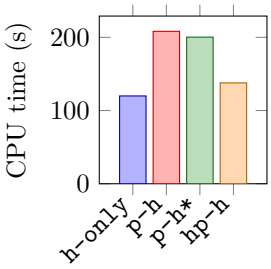
| Strategy | Coarsening | Levels | it | $\varrho$ |
|---|---|---|---|---|
| `h-only` | 2 $h$- | 3 | 12 | 0.11 |
| `p-h` | 2 $p$-, 2 $h$- | 5 | 12 | 0.09 |
| `p-h*` | 2 $p$-, 2 $h$- | 5 | 11 | 0.08 |
| `hp-h` | 2 $hp$- | 3 | 12 | 0.12 |

(e) Additional information: type and number of coarsening operations performed, number of levels built, number of iterations of preconditioned FCG (it), asymptotic convergence rate ($\varrho$).

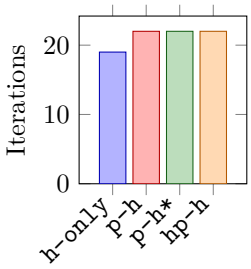
Figure 10: **Unit square, unstructured triangular mesh**, $k = 5$. The multigrid method is used as a **preconditioner** to FCG with the strategies of Table 1. Convergence is assumed achieved once the backward error reaches the tolerance of $10^{-12}$.



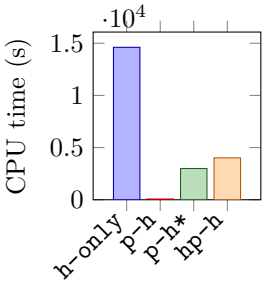

(a) Iteration comput. work




(b) Iteration time




(c) Number of iterations

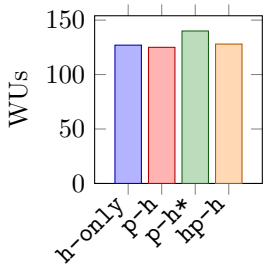



(d) Setup time

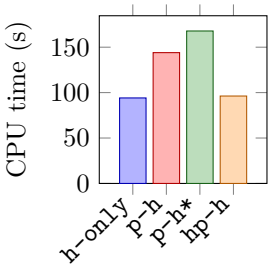
| Strategy | Coarsening | Levels | it | $\varrho$ |
|---|---|---|---|---|
| `h-only` | 4 $h$- | 5 | 19 | 0.27 |
| `p-h` | 2 $p$-, 4 $h$- | 7 | 22 | 0.40 |
| `p-h*` | 2 $p$-, 4 $h$- | 7 | 22 | 0.40 |
| `hp-h` | 4 $hp$- | 5 | 22 | 0.39 |

(e) Additional information: type and number of coarsening operations performed, number of levels built, number of iterations (it), asymptotic convergence rate ($\varrho$).

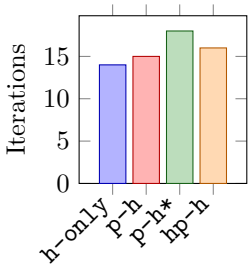
Figure 11: **Kellogg problem**, $k = 5$. The multigrid method is used **as a solver** with the strategies of Table 1. Convergence is assumed achieved once the backward error reaches the tolerance of $10^{-12}$.



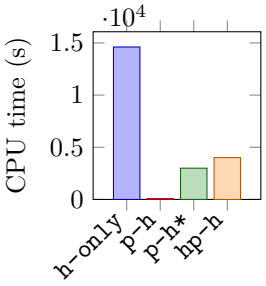

(a) Iteration comput. work




(b) Iteration time




(c) Number of iterations

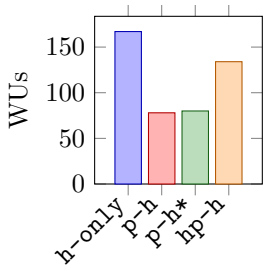



(d) Setup time

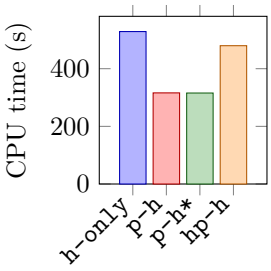
| Strategy | Coarsening | Levels | it | $\varrho$ |
|---|---|---|---|---|
| `h-only` | 4 $h$- | 5 | 14 | 0.14 |
| `p-h` | 2 $p$-, 4 $h$- | 7 | 15 | 0.18 |
| `p-h*` | 2 $p$-, 4 $h$- | 7 | 18 | 0.28 |
| `hp-h` | 4 $hp$- | 5 | 16 | 0.20 |

(e) Additional information: type and number of coarsening operations performed, number of levels built, number of iterations of preconditioned FCG (it), asymptotic convergence rate ($\varrho$).

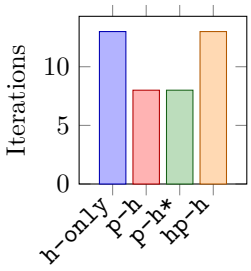
Figure 12: **Kellogg problem**, $k = 5$. The multigrid method is used as a **preconditioner** to FCG with the strategies of Table 1. Convergence is assumed achieved once the backward error reaches the tolerance of $10^{-12}$.



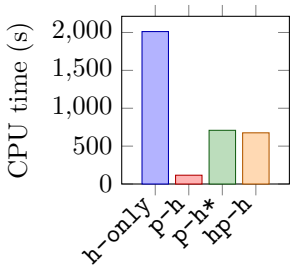

(a) Iteration comput. work




(b) Iteration time




(c) Number of iterations

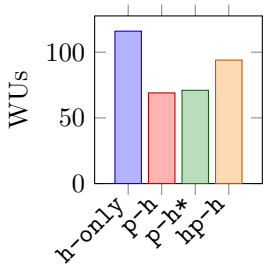



(d) Setup time

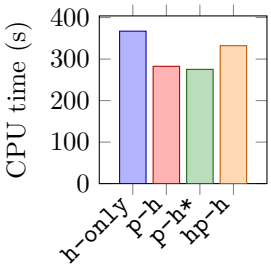
| Strategy | Coarsening | Levels | it | $\varrho$ |
|---|---|---|---|---|
| `h-only` | 4 $h$- | 5 | 13 | 0.10 |
| `p-h` | 1 $p$-, 4 $h$- | 6 | 8 | 0.04 |
| `p-h*` | 1 $p$-, 4 $h$- | 6 | 8 | 0.04 |
| `hp-h` | 2 $hp$-, 2 $h$- | 5 | 13 | 0.12 |

(e) Additional information: type and number of coarsening operations performed, number of levels built, number of iterations (it), asymptotic convergence rate ($\varrho$).

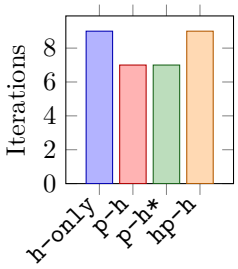
Figure 13: **Unit cube, Cartesian mesh** of $64^3$ elements, $k = 3$. The multigrid method is used **as a solver** with the strategies of Table 1. Convergence is assumed achieved once the backward error reaches the tolerance of $10^{-12}$.



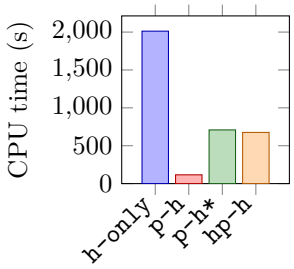

(a) Iteration comput. work




(b) Iteration time




(c) Number of iterations

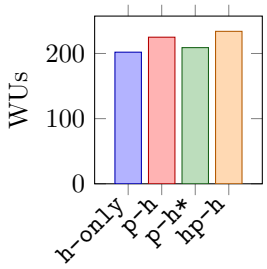



(d) Setup time

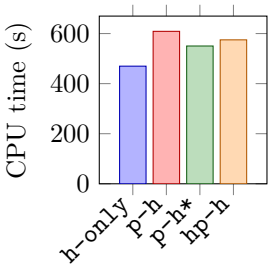
| Strategy | Coarsening | Levels | it | $\varrho$ |
|---|---|---|---|---|
| `h-only` | 4 $h$- | 5 | 9 | 0.03 |
| `p-h` | 1 $p$-, 4 $h$- | 6 | 7 | 0.02 |
| `p-h*` | 1 $p$-, 4 $h$- | 6 | 7 | 0.02 |
| `hp-h` | 2 $hp$-, 2 $h$- | 5 | 9 | 0.05 |

(e) Additional information: type and number of coarsening operations performed, number of levels built, number of iterations of preconditioned FCG (it), asymptotic convergence rate ($\varrho$).

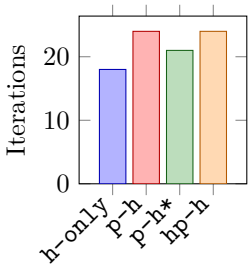
Figure 14: **Unit cube, Cartesian mesh** of $64^3$ elements, $k = 3$. The multigrid method is used **as a preconditioner to FCG** with the strategies of Table 1. Convergence is assumed achieved once the backward error reaches the tolerance of $10^{-12}$.



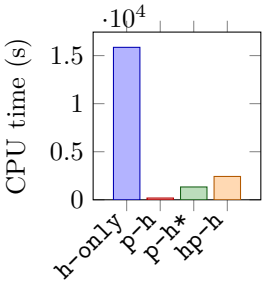

(a) Iteration comput. work




(b) Iteration time




(c) Number of iterations

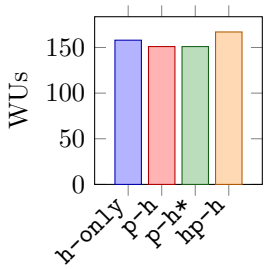



(d) Setup time

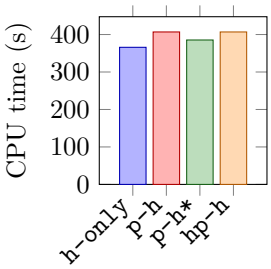
| Strategy | Coarsening | Levels | it | $\varrho$ |
|---|---|---|---|---|
| `h-only` | 4 $h$- | 5 | 18 | 0.26 |
| `p-h` | 1 $p$-, 4 $h$- | 6 | 24 | 0.35 |
| `p-h*` | 1 $p$-, 4 $h$- | 6 | 21 | 0.30 |
| `hp-h` | 2 $hp$-, 2 $h$- | 5 | 24 | 0.34 |

(e) Additional information: type and number of coarsening operations performed, number of levels built, number of iterations (it), asymptotic convergence rate ($\varrho$).

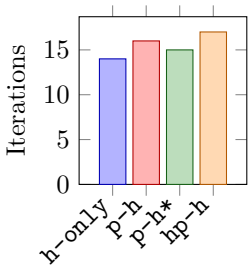
Figure 15: **Unit cube, structured tetrahedral mesh**, $k = 3$. The multigrid method is used **as a solver** with the strategies of Table 1. Convergence is assumed achieved once the backward error reaches the tolerance of $10^{-12}$.



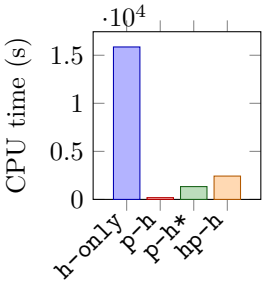

(a) Iteration comput. work




(b) Iteration time




(c) Number of iterations

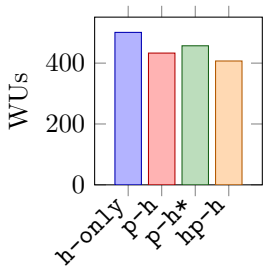



(d) Setup time

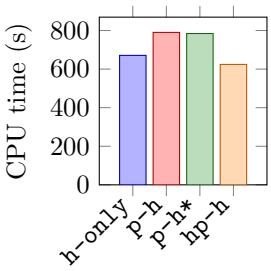
| Strategy | Coarsening | Levels | it | $\varrho$ |
|---|---|---|---|---|
| `h-only` | 4 $h$- | 5 | 14 | 0.16 |
| `p-h` | 1 $p$-, 4 $h$- | 6 | 16 | 0.19 |
| `p-h*` | 1 $p$-, 4 $h$- | 6 | 15 | 0.17 |
| `hp-h` | 2 $hp$-, 2 $h$- | 5 | 17 | 0.21 |

(e) Additional information: type and number of coarsening operations performed, number of levels built, number of iterations of preconditioned FCG (it), asymptotic convergence rate ($\varrho$).

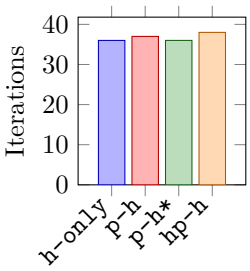
Figure 16: **Unit cube, structured tetrahedral mesh**, $k = 3$. The multigrid method is used **as a preconditioner to FCG** with the strategies of Table 1. Convergence is assumed achieved once the backward error reaches the tolerance of $10^{-12}$.



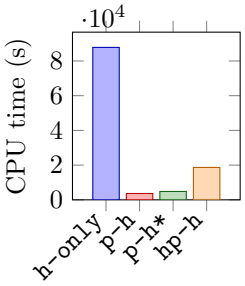

(a) Iteration comput. work




(b) Iteration time




(c) Number of iterations




(d) Setup time

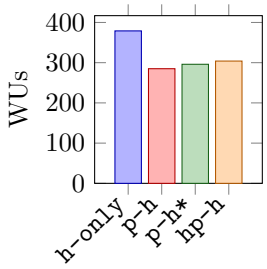
| Strategy | Coarsening | Levels | it | $\varrho$ |
|---|---|---|---|---|
| `h-only` | 5 *h*- | 6 | 36 | 0.59 |
| `p-h` | 1 *p*-, 5 *h*- | 7 | 37 | 0.60 |
| `p-h*` | 1 *p*-, 5 *h*- | 7 | 36 | 0.58 |
| `hp-h` | 1 *hp*-, 4 *h*- | 6 | 38 | 0.59 |

(e) Additional information: type and number of coarsening operations performed, number of levels built, number of iterations (it), asymptotic convergence rate ($\varrho$).

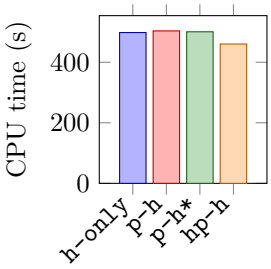
| Mesh | Elements | Coarsening factor |
|---|---|---|
| 1 | 311,755 | - |
| 2 | 46,375 | 1.82 |
| 3 | 8566 | 2.02 |
| 4 | 2515 | 1.35 |
| 5 | 1704 | 1.31 |
| 6 | 1552 | 1.71 |

(f) Mesh hierarchy. The coarsening factors are computed with respect to the mesh sizes.

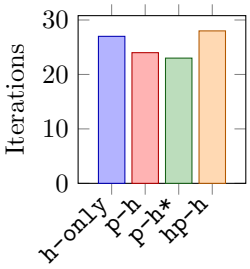
Figure 17: **Geneva wheel, non-nested unstructured meshes**, $k = 2$. The multigrid method is used **as a solver** with the strategies of Table 1. Convergence is assumed achieved once the backward error reaches the tolerance of $10^{-12}$.



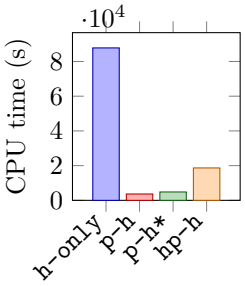

(a) Iteration comput. work




(b) Iteration time




(c) Number of iterations




(d) Setup time

| Strategy | Coarsening | Levels | it | $\varrho$ |
|---|---|---|---|---|
| `h-only` | 5 *h*- | 6 | 27 | 0.44 |
| `p-h` | 1 *p*-, 5 *h*- | 7 | 24 | 0.40 |
| `p-h*` | 1 *p*-, 5 *h*- | 7 | 23 | 0.37 |
| `hp-h` | 1 *hp*-, 4 *h*- | 6 | 28 | 0.45 |

(e) Additional information: type and number of coarsening operations performed, number of levels built, number of iterations of preconditioned FCG (it), asymptotic convergence rate ($\varrho$).

Figure 18: **Geneva wheel, non-nested unstructured meshes**, $k = 2$. The multigrid method is used **as a preconditioner to FCG** with the strategies of Table 1. Convergence is assumed achieved once the backward error reaches the tolerance of $10^{-12}$.